Department of Mechanical Engineering & Aeronautics
Division of Energy, Aeronautics & Environment
Laboratory of Computational Fluid Dynamics

Diploma Thesis
A Computational Study of Blood Flow in the Left Atrium of Patients with Atrial Fibrillation

Pavlos I. Varsos
1059722
Supervisor: Papadopoulos Polycarpos

Patras, July 2022
“ἐὰν μὴ ἐλπηται ανέλπιστον οὐκ ἐξευρήσει, ανεξερεύνητον ἐὸν καὶ ἄπορον”

– Heraclitus (544-484 B.C.)

To my parents, Maria and Giannis,

and my brothers, Giorgos and Alexandros
The present thesis was presented
by
Pavlos I. Varsos
1059722

On the 13th of July 2022
The approval of the thesis does not mean the acceptance of the writer’s views. During the authorship the principles of academic ethics have been respected.
Περίληψη

Η παρούσα Διπλωματική Εργασία έχει ως βασικό αντικείμενο την υπολογιστική μελέτη της ροής αίματος στον αριστερό κόλπο της καρδιάς ασθενών με Κολπική Μαρμαρυγή. Η Κολπική Μαρμαρυγή συνιστά την συχνότερη καρδιακή αρρυθμία παγκοσμίως, καθώς επηρεάζει το 1 με 2% του γενικού πληθυσμού, ενώ παράλληλα συνοδεύεται από υψηλά ποσοστά θνητότητας. Η Κολπική Μαρμαρυγή, όπως και κάθε αρρυθμία, ουσιαστικά αναφέρεται στις περιπτώσεις κάτω από τις οποίες η καρδιακή ηλεκτρική δραστηριότητα παρουσιάζει ανωμαλίες, οι οποίες συχνά δεν αντιμετωπίζονται πλήρως, ούτε φαρμακευτικά, ούτε χειρουργικά. Βασική μέθοδος, ελάχιστα επεμβατική, με στόχο την αποσυμφόρηση του ασθενή είναι η κατάλυση με καθετήρα, η οποία συνοδεύεται από ηλεκτροφυσιολογική χαρτογράφηση του αριστερού κόλπου. Στόχος της εργασίας είναι η δημιουργία ενός υποβάθρου για την σύγκριση χαρτογραφήσεων διάφορων δεικτών της τοιχωματικής διατμητικής τάσης (η οποία σύμφωνα με πειράματα συνδέεται με την καρδιακή ηλεκτρική δραστηριότητα) με ηλεκτροφυσιολογικές χαρτογράφησεις, παρεχόμενες από γιατρούς για τον κάθε ασθενή έξωθε τεχνικά. Πιθανές ομοιότητες δύναται να προσφέρουν σημαντικά δεδομένα στους γιατρούς προς μια κατεύθυνση για αύξηση της αποδοτικότητας της διαδικασίας της κατάλυσης. Για την δημιουργία του τρισδιάστατου μοντέλου του αριστερού κόλπου χρησιμοποιούνται μέθοδοι ανακατασκευής από δισδιάστατες εικόνες μικροοπτικής τομογραφίας. Μετά από μια διαδικασία βελτιστοποίησης του μοντέλου με διάφορες τεχνικές, ακολουθεί η πλεγματοποιήση του και εν συνεχεία η λύση της αιμοδυναμικής ροής. Οι δύο αυτές διαδικασίες πραγματοποιούνται με το λογισμικό ανοιχτού κώδικα OpenFOAM®, εκμεταλλεύοντας τις αμέτρητες δυνατότητες που παρέχουν τέτοιο τύπου λογισμικά. Την πλεγματοποίηση του μοντέλου, έπειτα από τη ρύθμιση των κατάλληλων παραμέτρων, ακολουθεί η μελέτη συναρμολογήσεων του πλεγματοσκευασμάτος, με στόχο την κατάλυση της καταλληλότητας του μοντέλου, η οποία και διεξάγεται σε συνθήκες μόνιμης ροής με τη χρήση του επιλύτη simpleFoam. Παράλληλα, ελέγχονται διάφορα μοντέλα τύρβης και συνοριακές συνθήκες με στόχο την κατάλυση πιθανών διαφορών στη ανώτερη αιμοδυναμική τάση του αίματος. Τέλος, πραγματοποιείται προσομοίωση της ροής υπό συνθήκες υπολογιστικού καρδιακού κόλπου με τη χρήση του επιλύτη pisoFoam, έτσι ώστε να καταλήξουμε στην απεικόνιση της κατανομής διάφορων αιμοδυναμικών δεικτών για τον αριστερό κόλπο, όπως η μεταβαλλόμενη διατμητική τάση καθώς και δείκτες που αφορούν τη πιθανότητα θρομβογένεσης σε περιοχές υψηλού κινδύνου, όπως το ιότο του αριστερού κόλπου.

Λέξεις Κλειδιά

Υπολογιστική Ρευστοδυναμική, OpenFOAM, Τοιχωματική Διατμητική Τάση, Κολπική Μαρμαρυγή, Αιμοδυναμική
Abstract

The main object of this Diploma Thesis is the computational study of blood flow in the left atrium of patients with Atrial Fibrillation. Atrial Fibrillation is the most frequent cardiac arrhythmia worldwide, since it affects 1 to 2% of the general population, while it is accompanied by high mortality rates. Like any other arrhythmia, Atrial Fibrillation refers to cases in which the electrical activity of the heart involves abnormalities, which are often not completely treated with neither medication nor surgery. A critical treatment method, minimally invasive, is catheter ablation which is simultaneously conducted with the procedure of electrophysiological mapping of the left atrium. The scope of this project is to create a framework for the comparison of mappings of multiple wall shear stress indices (which according to experiments are associated with the cardiac electrical activity) with electrophysiological mappings, using patient-specific data provided by doctors. Possible correlations can supply doctors with crucial information in a direction of improving the ablation process outcomes. Reconstruction techniques are implemented to convert two-dimensional MRI images into three-dimensional models of the left atrium. Following the optimization of the 3D model, the mesh generation and the solution of the cardiac flow procedures take place. Both of these processes are performed in the open-source software, OpenFOAM®, as we take advantage of the countless capabilities offered. After generating the mesh, a procedure of paramount importance for a computational study, the mesh independence study is carried out under steady state conditions using the simpleFoam solver, aiming to obtain the most suitable model for the present project. At the same time, various turbulence models and boundary conditions are tested in order to detect any possible differences in blood flow behavior. Finally, a transient flow simulation of a realistic cardiac cycle is conducted, utilizing the pisoFoam solver, since our main goal is to visualize the distribution of a variety of hemodynamic parameters, such as the Oscillatory Shear Index, as well as indices for indicating blood stasis in high-risk areas of the left atrium, like the Left Atrial Appendage.

Keywords

CFD, OpenFOAM, Wall Shear Stress, Atrial Fibrillation, Hemodynamics
Contents

List of Figures .................................................................................................................. xii
List of Tables .................................................................................................................... xv
Symbols & Abbreviations ............................................................................................... xvi
Preface ............................................................................................................................... xviii
1 Introduction .................................................................................................................. 1
  1.1 Purpose of Thesis ...................................................................................................... 1
  1.2 Thesis’ Structure ...................................................................................................... 1
2 The cardiovascular system ............................................................................................ 3
  2.1.1 The Heart ............................................................................................................. 4
  2.1.2 Blood Vessels ..................................................................................................... 10
  2.1.3 Blood .................................................................................................................. 12
  2.2 The Left Atrium ...................................................................................................... 12
  2.3 Atrial Fibrillation .................................................................................................... 14
    2.3.1 Definition of Cardiac Arrhythmias ...................................................................... 14
    2.3.2 Atrial Fibrillation: definition, causes and management ........................................ 14
3 Fluid Mechanics & Hemodynamics .............................................................................. 19
  3.1 Fluid Mechanics ..................................................................................................... 19
    3.1.1 Properties of fluids .......................................................................................... 19
    3.1.2 Laminar and Turbulent Flow ........................................................................... 19
    3.1.3 Entrance Region and Entry Lengths ................................................................. 21
    3.1.4 Governing equations of fluid motion ................................................................. 22
    3.1.5 Exact solutions to Navier-Stokes Equations ...................................................... 24
    3.1.6 A note on turbulent flows ................................................................................. 25
    3.1.7 Vortex Dynamics ............................................................................................... 26
  3.2 Hemodynamics and key hemodynamic indices ....................................................... 28
    3.2.1 Compliance ....................................................................................................... 28
    3.2.2 Resistance ........................................................................................................ 29
    3.2.3 Wall Shear Stress ............................................................................................. 30
    3.2.4 Time Averaged Wall Shear Stress ..................................................................... 31
    3.2.5 Oscillatory Shear Index .................................................................................... 31
    3.2.6 Endothelial Cell Activation Potential ............................................................... 32
3.2.7  Relative Residence Time ................................................................. 32

4  Computational Fluid Dynamics - Hemodynamics ............................................ 33
  4.1  Space discretization and numerical schemes ........................................... 33
    4.1.1  Finite Differences Method ............................................................. 34
    4.1.2  Finite Volume Method ................................................................. 36
  4.2  Computational Hemodynamics ............................................................... 39
    4.2.1  Advantages of Computational Hemodynamics ...................................... 40
    4.2.2  Computational Modeling of Cardiac Hemodynamics ............................. 40
  4.3  OpenFOAM ............................................................................................ 41
    4.3.1  Solvers .............................................................................................. 42
    4.3.2  Standard “Utilities” ........................................................................... 42
    4.3.3  File structure of OpenFOAM cases ..................................................... 42
    4.3.4  The numerical algorithms: SIMPLE, PISO and PIMPLE ...................... 45

5  Pre-Processing .............................................................................................. 47
  5.1  Reconstruction of the 3D model ............................................................... 47
  5.2  Meshing Procedure .................................................................................. 51
    5.2.1  BlockMesh and surfaceFeatureExtract Utilities .................................... 51
    5.2.2  SnappyHexMesh Utility and parallel run ............................................. 53
    5.2.3  Definition of Patches – topoSet and createPatch Utilities ..................... 61
    5.2.4  CheckMesh Utility ............................................................................. 64
    5.2.5  Mesh generation in Ansys Fluent (Academic Version) ........................... 65

6  Solution Method ............................................................................................ 69
  6.1  Solvers and Numerical Schemes .............................................................. 69
  6.2  Mesh Independence Study ....................................................................... 72
    6.2.1  Turbulence model selection ............................................................... 72
    6.2.2  LES boundary conditions ................................................................. 74
    6.2.3  First Case Study ................................................................................ 75
    6.2.4  Second Case Study .......................................................................... 78
    6.2.5  Effect of inaccuracies in mitral flow on the WSS distribution ............... 79
    6.2.6  Comparison between LES, k-Epsilon, k-OmegaSST turbulence models .... 80
  6.3  Transient Flow Simulation ....................................................................... 85

7  Results ......................................................................................................... 87
8 Conclusion................................................................................................................................. 95
  8.1 Limitations ............................................................................................................................ 96
  8.2 Future work .......................................................................................................................... 96
9 References.................................................................................................................................. 98
Appendix I ..................................................................................................................................... 107
Appendix II .................................................................................................................................... 111
Appendix III ................................................................................................................................... 112
List of Figures

Figure 2.1. A complete diagram of the human heart anatomy [2].......................................................... 3
Figure 2.2. Anatomy of the mitral valve [5].......................................................................................... 6
Figure 2.3. The conducting pathway of the human heart [7]................................................................. 7
Figure 2.4. A normal ECG with its phases [2].................................................................................... 7
Figure 2.5. Events of the cardiac cycle for ventricular function – Wiggers Diagram [4]............. 8
Figure 2.6. Typical right atrial pressures: a-wave = 2.5-7 mmHg; v-wave = 2-7.5 mmHg; Mean pressure = 2-5 mmHg [10].......................................................... 9
Figure 2.7. Typical left atrial pressures: a-wave = 4-16 mmHg; v-wave = 6-21 mmHg; Mean pressure = 2-12 mmHg [10]....................................................... 9
Figure 2.8. Blood volume distribution among the vessels [4]............................................................. 10
Figure 2.9. Pressure fluctuations in different parts of the cardiovascular system [4]................. 11
Figure 2.10. MAP and Pulse Pressure [11]....................................................................................... 11
Figure 2.11. Blood viscosity vs. hematocrit. Sharp increase observed in blood viscosity after HCT>40 [12]........................................................................... 12
Figure 2.12. Different locations of Left Atrial Geometry [19]............................................................ 13
Figure 2.13. Left atrial pressure-volume loop demonstrating left atrial reservoir, conduit, and pump functions. MV/O-C refers to opening and closure of MV [20]................................. 13
Figure 2.14. Atrial Fibrillation vs normal sinus rhythm. Disorganization of the signals in atria during AF [18] ........................................................................................................ 14
Figure 2.15. Normal and reentrant excitation pathways [14]........................................................... 15
Figure 2.16. Locations of ectopic foci found in a sample of patients by Haïssaguerre et al.(1998) .......................................................................................................................... 16

Figure 2.17. Catheter ablation (cryoablation and radiofrequency current) procedure for the isolation of the pulmonary veins [35]............................................................................. 17
Figure 2.18. Left atrium voltage-electrophysiological mapping [33]............................................... 18
Figure 2.19. Left Atrial Appendage Closure Device (Watchman device by Boston Scientific) [37]...................................................................................................................... 18
Figure 3.1. Comparison of laminar and turbulent flow [39]............................................................ 20
Figure 3.2. Comparison of laminar and turbulent flow velocity profiles across a pipe section [42] ............................................................................................................................ 21

Figure 3.3. The development of the velocity boundary layer in a pipe [40]........................................ 21
Figure 3.4. Infinitesimally small fluid element and forces in the x direction only [43]............. 22
Figure 3.5. Hagen-Poiseuille Flow [42]............................................................................................. 24
Figure 3.6. Velocity fluctuations under turbulent flow [38]............................................................ 25
Figure 3.7. Transmitral vortex formation in a patient with severe mitral stenosis. Blue and red represent opposite rotational directions [63]........................................................................... 28
Figure 3.8. Compliance curve for a biological tissue such as artery [11]...................................... 28
Figure 3.9. Normal and hypertrophied heart differences in pressure-volume relations [42].. 29
Figure 3.10. Dependence of flowrate F (Q) on vessel radius [11]................................................. 30
Figure 3.11. Three different ways of velocity computation/measurement. (a) refers to Phase Contrast MRI in the aortic valve region [46], (b) shows the intravascular ultrasound procedure [47], and (c) is an example of applied CFD in a cerebral aneurysm ............................................ 31
Figure 4.1. Discrete grid points [43]........................................................................33
Figure 4.2. Graphical representation of a 2D and 3D grid containing cells [55]........37
Figure 4.3. Cell-Centered and Vertex-Centered FV methods [56]............................37
Figure 4.4. Structured and unstructured grid [57].....................................................37
Figure 4.5. Most widely used element types [54]......................................................38
Figure 4.6. Aspect ratio and skewness of 2D elements ............................................38
Figure 4.7. Scientific Fields required for a Computational Hemodynamics study. Idea derived from [58]........................................................................................................39
Figure 4.8. Three applications of Computational Hemodynamics. (A) refers to coronary stent design [60], (B) is a ventricular assist device [61], and (C) is a simulation in a human aorta [62].................................................................40
Figure 4.9. Overview of OpenFOAM structure [70]..................................................41
Figure 4.10. icoFoam solver source code ..................................................................44
Figure 4.11. CreateFields source code ....................................................................45
Figure 5.1. 3D model reconstruction from Phase-Contrast Magnetic Resonance Imaging, MRI sample image obtained from [73]........................................................................................................47
Figure 5.2. 3D slicer interface, and our geometry after cutting the non-uniform inlets .....48
Figure 5.3. Smoothing the geometry with Blender ....................................................48
Figure 5.4. User Interface of VMTK toolkit ..................................................................49
Figure 5.5. The clipping procedure ..........................................................................49
Figure 5.6. Added flow extensions (L/D =20) to the inlets .........................................50
Figure 5.7. Final form of the LA geometry ..................................................................51
Figure 5.8. BlockMesh Dictionary ............................................................................52
Figure 5.9. Generation of the block containing the geometry ...................................52
Figure 5.10. Mesh generated with castellatedMeshControls only turned on .............54
Figure 5.11. Mesh generated with both castellatedMeshControls and SnapControls turned on . 55
Figure 5.12. Boundary layer parameters set on addLayers section [76] .....................56
Figure 5.13. Irregularities fixed by changing featureAngle value ...............................57
Figure 5.14. Irregularities at the patches section caused by the featureAngle value ....57
Figure 5.15. Difference between non-orthogonal and skew mesh [78].......................58
Figure 5.16. DecomposePar Dictionary ....................................................................59
Figure 5.17. Decompose of the initial domain into smaller parts, each one corresponding to one processor .................................................................................................59
Figure 5.18. Initiation of the parallel run ....................................................................60
Figure 5.19. Complete guide for mesh generation including all the necessary dictionaries and utilities.............................................................................................................61
Figure 5.20. TopoSet dictionary sample .....................................................................62
Figure 5.21. Definition of an inlet with topoSet .........................................................63
Figure 5.22. Printed face normal vectors for RSPV inlet ............................................63
Figure 5.23. Final definition of patches ....................................................................64
Figure 5.24. Mitral valve patch, and internal mesh including 4 boundary layers ........64
Figure 5.25. Use of cylinders and bounding box for the shrinkwrap process in Ansys Fluent meshing tool ........................................................................................................66
Figure 5.26. Polyhedral Mesh generated using Ansys Fluent Meshing tool ........................................... 67
Figure 6.1. fvSchemes dictionary for specifying numerical schemes......................................................... 70
Figure 6.2. fvSolution dictionary. Includes matrix solvers, tolerances, etc.............................................. 71
Figure 6.3. Comparison between the initial residuals of P,U vs iterations using different relaxation factors in a steady-state simulation. Right figure shows the divergence in the initial residuals ................................................................................................................................. 72
Figure 6.4. Comparison between the 3 main turbulence models. Idea from [82]........................................ 73
Figure 6.5. LES dynamic k eqn source file.................................................................................................. 74
Figure 6.6. Mitral valve velocity in normal rhythm and AFib A wave is not present in AFib, indicating absence of atrial contraction ................................................................................................................................. 75
Figure 6.7. Flow conditions in a cross section of meshes 2 and 3 ............................................................. 76
Figure 6.8. LIPV and LSPV inlets’ profile using mesh 1 .............................................................................. 77
Figure 6.9. LIPV and LSPV inlets’ profile using mesh 2 .............................................................................. 77
Figure 6.10. LIPV and LSPV inlets’ profile using mesh 3 ............................................................................ 77
Figure 6.11. Bar graph of WSS values versus number of boundary layers .............................................. 78
Figure 6.12. WSS distribution among 5 models .......................................................................................... 79
Figure 6.13. WSS distribution comparison between -20% and +20% difference in MV velocity .......... 80
Figure 6.14. Initial residuals of Ux compared using LES, k-Epsilon and k-OmegaSST models 82
Figure 6.15. The completely parabolic profile implemented on LIPV and LSPV inlets......................... 82
Figure 6.16. High velocity flow hitting the LA wall resulting in an increase in WSS ......................... 83
Figure 6.17. WSS distribution using the LES model .................................................................................. 83
Figure 6.18. WSS distribution using the k-OmegaSST model ................................................................. 84
Figure 6.19. WSS distribution using the k-Epsilon model ........................................................................ 84
Figure 6.20. Lambda2 criterion source file .............................................................................................. 86
Figure 7.1. LIPV and LSPV inlets’ velocity profile under maximal atrial contraction .............................. 87
Figure 7.2. LIPV and LSPV inlets’ velocity profile under MV reverse flow ............................................ 87
Figure 7.3. Blood flow patterns under MV reverse flow .......................................................................... 88
Figure 7.4. Blood flow patterns under maximal atrial contraction .......................................................... 88
Figure 7.5. Vortex structures detected under maximal atrial contraction using lambda2 criterion .......... 89
Figure 7.6. Vortex structures detected under MV reverse flow using lambda2 criterion ..................... 89
Figure 7.7. Wall Shear Stress distribution on the LA wall under MV reverse flow .............................. 90
Figure 7.8. Wall Shear Stress distribution on the LA wall under maximal atrial contraction............ 91
Figure 7.9. Time Averaged Wall Shear Stress distribution ................................................................. 91
Figure 7.10. Oscillatory Shear Index distribution ................................................................................... 92
Figure 7.11. Endothelial Cell Activation Potential distribution ............................................................ 93
Figure 7.12. Relative Residence Time distribution ................................................................................. 93
List of Tables

Table 2.1. Approximate quantification of individual vessels in the circulatory system .......... 10
Table 4.1. Numerical methods most commonly used for the solution of elliptic, parabolic and hyperbolic PDEs ................................................................. 36
Table 4.2. A sample of standard solvers used in different kinds of problems. Solvers in bold will be used in the present work ........................................................................ 42
Table 5.1. Main parameters for a good quality mesh ................................................................. 60
Table 5.2. Mesh details after running checkMesh utility ............................................................ 65
Table 5.3. Mesh details after running checkMesh utility for Ansys Fluent generated mesh ...... 67
Table 6.1. 1st part of mesh independence study ........................................................................ 76
Table 6.2. Specifying the appropriate number of elements ....................................................... 78
Table 6.3. Estimation of inaccuracies in WSS distribution ......................................................... 79
Table 6.4. Area, Diameter, Flow rate, velocity, and Reynolds number for every inlet .......... 81
Table 6.5. Boundary Conditions for transient flow study .......................................................... 85
## Symbols & Abbreviations

### Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{\text{mean}} )</td>
<td>Blood Mean Pressure</td>
</tr>
<tr>
<td>( P_{\text{systole}} )</td>
<td>Systolic Blood Pressure</td>
</tr>
<tr>
<td>( P_{\text{diastole}} )</td>
<td>Diastolic Blood Pressure</td>
</tr>
<tr>
<td>( \nu )</td>
<td>Kinematic Viscosity</td>
</tr>
<tr>
<td>( \tau )</td>
<td>Shear Stress</td>
</tr>
<tr>
<td>( \mu )</td>
<td>Dynamic Viscosity</td>
</tr>
<tr>
<td>( \rho )</td>
<td>Density of the fluid</td>
</tr>
<tr>
<td>( u, v, w )</td>
<td>Velocity components</td>
</tr>
<tr>
<td>( \text{Re} )</td>
<td>Reynolds Number</td>
</tr>
<tr>
<td>( D )</td>
<td>Diameter</td>
</tr>
<tr>
<td>( L_h )</td>
<td>Hydrodynamic Entry Length</td>
</tr>
<tr>
<td>( \nabla V )</td>
<td>Velocity Divergence</td>
</tr>
<tr>
<td>( \sigma )</td>
<td>Normal Stress</td>
</tr>
<tr>
<td>( T_{x,y,z} )</td>
<td>Shear Force</td>
</tr>
<tr>
<td>( \Delta p )</td>
<td>Pressure Difference</td>
</tr>
<tr>
<td>( \omega )</td>
<td>Vorticity vector</td>
</tr>
<tr>
<td>( Q^* )</td>
<td>Second invariant of ( \nabla u )</td>
</tr>
<tr>
<td>( J )</td>
<td>Velocity gradient tensor</td>
</tr>
<tr>
<td>( C )</td>
<td>Compliance</td>
</tr>
<tr>
<td>( E )</td>
<td>Elastance</td>
</tr>
<tr>
<td>( Q )</td>
<td>Flow rate</td>
</tr>
<tr>
<td>( R )</td>
<td>Resistance</td>
</tr>
<tr>
<td>( t )</td>
<td>Time</td>
</tr>
<tr>
<td>( T )</td>
<td>Temperature</td>
</tr>
<tr>
<td>( \alpha )</td>
<td>Thermal diffusivity</td>
</tr>
<tr>
<td>( \text{Co} )</td>
<td>Courant Number</td>
</tr>
<tr>
<td>( \varepsilon )</td>
<td>Turbulence dissipation rate</td>
</tr>
<tr>
<td>( \omega^* )</td>
<td>Turbulence specific dissipation rate</td>
</tr>
</tbody>
</table>

### Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AF</td>
<td>Atrial Fibrillation</td>
</tr>
<tr>
<td>A-V</td>
<td>Atrioventricular Valves</td>
</tr>
<tr>
<td>BC</td>
<td>Boundary Condition</td>
</tr>
<tr>
<td>CFD</td>
<td>Computational Fluid Dynamics</td>
</tr>
<tr>
<td>CFL</td>
<td>Courant Friedrichs Lewy</td>
</tr>
<tr>
<td>CHD</td>
<td>Computational Hemodynamics</td>
</tr>
<tr>
<td>CO</td>
<td>Cardiac Output</td>
</tr>
<tr>
<td>DNS</td>
<td>Direct Numerical Simulation</td>
</tr>
<tr>
<td>ECAP</td>
<td>Endothelial Cell Activation Potential</td>
</tr>
<tr>
<td>ECG</td>
<td>Electrocardiogram</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>EDV</td>
<td>End Diastolic Volume</td>
</tr>
<tr>
<td>ESV</td>
<td>End Systolic Volume</td>
</tr>
<tr>
<td>FEA</td>
<td>Finite Element Analysis</td>
</tr>
<tr>
<td>FSI</td>
<td>Fluid Structure Interaction</td>
</tr>
<tr>
<td>FVM</td>
<td>Finite Volume Method</td>
</tr>
<tr>
<td>HR</td>
<td>Heart Rate</td>
</tr>
<tr>
<td>LA</td>
<td>Left Atrium</td>
</tr>
<tr>
<td>LAA</td>
<td>Left Atrial Appendage</td>
</tr>
<tr>
<td>LAVI</td>
<td>Left Atrial Volume Index</td>
</tr>
<tr>
<td>LES</td>
<td>Large Eddy Simulation</td>
</tr>
<tr>
<td>LV</td>
<td>Left Ventricle</td>
</tr>
<tr>
<td>MAP</td>
<td>Mean Arterial Pressure</td>
</tr>
<tr>
<td>MRI</td>
<td>Magnetic Resonance Imaging</td>
</tr>
<tr>
<td>MV</td>
<td>Mitral Valve</td>
</tr>
<tr>
<td>N-S</td>
<td>Navier Stokes</td>
</tr>
<tr>
<td>OpenFOAM</td>
<td>Open-source Field Operation And Manipulation</td>
</tr>
<tr>
<td>OSI</td>
<td>Oscillatory Shear Index</td>
</tr>
<tr>
<td>PDE</td>
<td>Partial Differential Equation</td>
</tr>
<tr>
<td>PISO</td>
<td>Pressure Implicit of Split Operations</td>
</tr>
<tr>
<td>PV</td>
<td>Pulmonary Vein(s)</td>
</tr>
<tr>
<td>RA</td>
<td>Right Atrium</td>
</tr>
<tr>
<td>RANS</td>
<td>Reynolds Averaged Navier Stokes</td>
</tr>
<tr>
<td>RRT</td>
<td>Relative Residence Time</td>
</tr>
<tr>
<td>RV</td>
<td>Right Ventricle</td>
</tr>
<tr>
<td>SA</td>
<td>Sinoatral</td>
</tr>
<tr>
<td>SIMPLE</td>
<td>Semi Implicit Method of Pressure Linked Equations</td>
</tr>
<tr>
<td>SV</td>
<td>Stroke Volume</td>
</tr>
<tr>
<td>TAWSS</td>
<td>Time Averaged Wall Shear Stress</td>
</tr>
<tr>
<td>WSS</td>
<td>Wall Shear Stress</td>
</tr>
</tbody>
</table>
Preface

The present thesis is part of a project between École Polytechnique Fédérale de Lausanne (EPFL) and Université de Genève in which I had the honor to participate during my internship in the Laboratory of Hemodynamics and Cardiovascular Technology (LHTC) during the summer of 2021. I would like to express my gratitude to Professor Nikolaos Stergiopulos for welcoming me in his lab and allowing me to continue working in this project as the main object of my thesis. The continuous assistance provided by George Rovas, PhD student at LHTC, was greatly appreciated.

Furthermore, I wish to show my appreciation to my supervisor at University of Patras, Professor Polycarpos Papadopoulos, for his constant guidance since our very first meeting back in November 2020. His willingness to support me in the fascinating and promising scientific field of Computational Fluid Dynamics, and most importantly that of Hemodynamics, has led me to where and who I am today.

Last but not least, I would like to thank my family for providing me with both psychological and financial support during my five years of study in Patras. Their unwavering support and belief in me have shaped my personality, pushing me every day towards into a better version of myself. I ought to thank my brothers, George for the inspiration and Alex for his apt remarks on cardiology topics of my thesis.
1 Introduction

1.1 Purpose of Thesis
The present thesis is a compulsory subject in the curriculum of Mechanical Engineering & Aeronautics Department at University of Patras and was conducted in the Academic Year 2021-2022.

Our study is founded on the principles of Computational Fluid Dynamics. Computational techniques have been implemented extensively the last two decades in a direction of providing the medical society with useful diagnostic tools and realistic solutions. As far as the cardiovascular system is concerned, a great deal of models have been developed aiming to study the blood flow patterns in different regions of it and investigate possible associations of the flow with specific diseases. Typical cases for which CFD methods have been applied to, are the wall shear stress distributions in aneurysm geometries, for detecting the risk of a possible rupture of the aneurysm sac, and in the carotid artery branches for studying the mechanism of arteriosclerosis. However, scientists do not always create 3D models. Reduced order models, such as 0-D and 1-D models, can provide exceptional information regarding the flow in the arterial tree of the human body, whereas for a complete 3D simulation, terrific amount of computing power is required. The advantage of 3D models in detecting detailed blood flow patterns, is due to the improvements in the MRI imaging field and the reduction in computing costs.

The current thesis aims to contribute to the advancements in Atrial Fibrillation treatment, which involve the catheter ablation procedure. Electrophysiological mappings of the human left atrium have made it possible to detect regions of interest, speaking of abnormalities in the electrical activity (e.g., voltage distribution) that are responsible for the initiation and maintenance of this type of arrhythmia. Recent studies have shown that the electrical activity of the heart is associated with its hydrodynamic environment, which is either the blood flow patterns within the left atrium or generally the wall shear stress distribution. Therefore, by acquiring patient specific data we focus on generating mappings of the different WSS indices distributions in the left atrium that will be then compared to the electrophysiological mappings. Possible associations between these 2 distributions will probably lead to a more extensive study of how WSS can act as a mechanism of Atrial Fibrillation and can also provide aid to physicians for a better targeted catheter ablation procedure, improving its outcomes. For the completeness of this thesis a thorough explanation of the computational techniques used will be conducted since for our study we take advantage of the potential an open-source software provides its users with. It’s needless to say that using an open-source software we get a complete idea of how the source code works giving us the capability of modifying it to our preferences, in contrast to the numerous commercial software for computational mechanics.

1.2 Thesis’ Structure
The current thesis consists of 9 chapters, from which the first and the last one, refer to the introduction and the references respectively. The second Chapter provides a thorough view of the required background for the conduction of this project, on basic cardiology matters, including the functionality of the heart under both normal and Atrial Fibrillation conditions. In the third Chapter, the main principles of fluid mechanics and hemodynamics are presented, involving the theoretical
background they rely on. The fourth Chapter focuses on the computational side of this thesis. A presentation of the main discretization methods is given, while the advantages of implementing CFD to applications in cardiovascular flows are also demonstrated. In addition, a complete analysis of OpenFOAM’s functionality is conducted.

The following chapters are those of the greatest importance in our study. Chapter 5 represents the pre-processing part of our study, since it contains the procedures of the reconstruction of the 3D model, as well as the mesh generation process. The sixth Chapter contains the whole solution method implemented, from the mesh independence study to the transient flow simulation. Comparisons using a variety of models and boundary conditions are carried out in order to detect possible differences, and how the latter affect the blood flow. Furthermore, in the seventh Chapter the final results of the transient flow are presented, including the visualization of the calculated WSS indices, and blood flow patterns in the left atrium. Finally, in Chapter 8, conclusion remarks are given., while limitations and possible improvements in our study are discussed.
The cardiovascular system

Before diving into the analysis of our project, it is of upmost importance to have a look into the theoretical background, which the present thesis depends on. For this reason, some of the vital functions of the human body will be presented, including mostly those taking place in our cardiovascular system.

The cardiovascular system is a network of organs (such as the lymphatic system, the digestive, etc.) which has caught the attention of thousands of engineers during the last decades, continuing to attract more and more, due to its severe complexity and the multiphysics of the mechanisms that it’s described from. Electrical signal processing, tissue engineering, solid mechanics, and fluid dynamics, are some of the scientific fields, that are required in order to study and simulate this system as a whole.

The human cardiovascular system is actually a closed loop system that consists of the heart, blood vessels and blood. Its main functions include the distribution of oxygen and nutrients to every single tissue of the human body through blood, the transportation of metabolic waste products (e.g. CO₂ and urea) from the tissues to the lungs and excretory organs and the distribution of water, electrolytes and hormones throughout the body as well. Moreover, it is responsible for maintaining homeostasis in the body, meaning that it contributes to the infrastructure of the immune system, while achieving thermoregulation [1]. In the following subchapters, a complete description of the three major components of the cardiovascular system will be given, starting with the most complex one.

![Figure 2.1. A complete diagram of the human heart anatomy [2]](image-url)
2.1.1 The Heart

2.1.1.1 Heart: Anatomy and Function

One of the most magnificent ever “made” machines, is, undoubtedly the human heart. The heart behaves as a hydraulic pump which keeps the blood flow constant in the human body. With an average mass of 275 g, it begins to beat after a few weeks of gestation, contracting approximately once per second through a lifetime (about 3 billion heart beats). The heart is enclosed within the pericardium, a thin but also tough connective tissue, that protects the heart within the chest cavity, while preventing the overfilling of the heart chambers [2]. Inside the pericardium sac, there is a fluid that acts as a lubricator when the heart contracts or generally moves within the space. The wall is actually made up of three different layers, going from the exterior to the interior, the epicardium, the myocardium and finally the endocardium.

Speaking of chambers, the heart consists of four of them, the left (main objective of this thesis) and right atria which receive the incoming blood from the venous circulation (see Chapter 2.2), and the left and right ventricles which eject the blood into the arterial circulation (Chapter 2.2).

The blood enters the right atrium through the superior and inferior vena cavae (known as the great veins), moves into the right ventricle and gets ejected to the lungs via the pulmonary arteries. The blood returning from the lungs, enters the left atrium via the pulmonary veins and then reaches its’ final chamber, the left ventricle, which massively ejects it through the aorta to every single part of the body. The blood’s path from the right ventricle to the lungs and then back to the left atrium is called pulmonary circulation, whereas, the circulation starting from the left ventricle, ending to the right atrium, is called systemic.

Both atria are small and have a thin wall compared to the ventricles. The pressure in the former, reaches up to only 10 mmHg (1 mmHg = 0.133 kPa), as, in order to “fill” the below to them ventricles, a minimal contraction of their walls is required. Regarding the right ventricle, the peak pressure value achieved is nearly 30 mmHg, which is enough to attain pulmonary circulation. In contrast, the left ventricle ejects blood at a much higher pressure, at around 120 mmHg, which is quite reasonable as blood is pumped to every part of the body, meaning that a higher pressure is required to overcome the resistance. This is the reason why the wall of the left ventricle is the thickest one (Figure 2.1).

Some standard quantities that are used to describe the heart function are the stroke volume and the cardiac output. At every heartbeat, 70 mL of blood is ejected from the left ventricle. This volume is known as stroke volume. Cardiac output (CO) is the product of stroke volume (SV) and heart rate (HR), so for a heartbeat close to 70 beats/min and a stroke volume of 70 mL, the cardiac output is approximately 5 L/min (Eq.(2.1)).

\[
CO = SV \cdot HR ,
\]

where, SV is the stroke volume (volume/beat) and HR, the heart rate (beats/min).

The stroke volume depends on the amount of blood in the ventricles’ chamber at the end of the filling phase, which is called end-diastolic volume (EDV). There is a strong connection of the previous with the contractile ability of the wall. In the left atrium, the EDV regularly ranges from
110 to 120 mL, while at the end of the contraction, we consider the end-systolic volume value (ESV) which ranges from 40 to 50 mL. Stroke volume (SV) is actually the difference between EDV and ESV, as described in Eq.(2.2).

\[ SV = EDV - ESV \] (2.2)

Another quantity that is commonly used, is the ejection fraction, which resembles the stroke volume as a percentage of the EDV. A normal range for the ejection fraction (EF) of males is between 52 and 72 % [3].

### 2.1.1.2 The Heart Valves

As can be noticed in Figure 2.1, there is no communication between the left and right atria, or the left and right ventricles. In the meantime, in order to prevent backflow of blood, the heart accommodates two sets of one-way valves, the atrioventricular valves (A-V) and the aortic/pulmonary (Semilunar) valves. The valves are either open or closed, depending on the pressure difference on either side of the valve.

The A-V set of valves consists of the tricuspid and mitral valve. Their role is to prevent backflow from the ventricles to the atria during systole. The tricuspid valve (three cusps/flaps) is located between the right atrium and ventricle, while the mitral (also called bicuspid, as it has two cusps), for which a special mention will be made afterwards, between the left atrium and ventricle. The semilunar valves, known as aortic and pulmonary valves, prevent backflow from the aorta and pulmonary arteries during diastole, therefore as someone can guess easily, the aortic valve is placed between the aorta and the left atrium, while the pulmonary, between the pulmonary artery and right ventricle. Atrioventricular valves are thin and filmy, compared to semilunar, as for the closure to occur, it’s required almost no backflow. The semilunar are much heavier and require a rapid backflow for a few milliseconds [4].

When the pressure in the atria is higher than in the ventricles, the A-V valves open and let blood flow through them. This takes place in the so-called diastole phase (relaxation phase), in which, the blood volume in the ventricles increase. At the end of the filling phase, the contraction of the ventricles starts (systole), resulting in a higher pressure which leads to the A-V valves closure. The closure of the valves is of great importance and that is why their structure is very robust. The valves are anchored to the walls of the ventricles (papillary muscle) by small tendon-like fibers, known as chordae tendinae, which attach to the valve cusps.

The semilunar valves function a bit differently from the A-V valves. The higher pressure in the arteries during the end of systole results to a less soft closure of these valves than the A-V ones. Moreover, due to their smaller area, the blood velocity is much higher than in the A-V valves, causing rapid closures and ejections, finally leading to a higher mechanical stress absorbed. Hence, their structure, differs as well compared to the atrioventricular one. Finally, it’s the closing of the valves, that causes the sounds heard with a stethoscope.

As the object of this thesis concerns the left atrium blood flow conditions, it would be reasonable to have an idea of the mitral valve morphology, keeping in mind that the mitral valve functions as the only outlet in the left atrium.
In Figure 2.2 [5] the anatomy of the mitral valve is illustrated. As can be noticed, the mitral valve is a complex structure on its own. It contains 2 leaflets (flaps or cusps), an annulus, tendinous cords and the papillary muscles that connect the valve leaflets with the ventricle wall. Normally the mitral valve has an area of around 4 to 6 cm$^2$ [6]. The two leaflets are called anteromedial (AVML) and posterolateral (PMVL). The anterior one covers the two thirds of the valve. Finally, the two leaflets are divided into eight segments (P1, P2, P3, AC, PC, A1, A2, A3).

Valve abnormalities are not that rare and could be a result of stenosis (stiff flaps, narrowing of the annulus) or the insufficiency in closing fully leading to backflow and blood leak into the atrium.

2.1.1.3 Electrical Activity of the Heart

As the present thesis deals with the flow conditions under atrial fibrillation (will be discussed in Chapter 2.4), it is of great significance to take a look into the theoretical background of the normal electrical activity of the heart. There are two kinds of cardiac muscle cells: the contractile cells which are responsible for creating the muscle force, and the autorhythmic cells, which consist of the pacemaker cells and the conducting system. The contraction of the heart is not a result of an outside signal but happens due to the action potential initiated in the pacemaker regions of the heart (the sinoatrial (SA) node and the atrioventricular (AV) node, Figure 2.3 [7]). The signal created travels through the conducting system (the bundle of His, and the Purkinje fiber, which also create action potentials). These pacemaker regions depolarize themselves at different rates. The S-A node is exciting at a rate of 70 beats/min, the A-V node at 40 beats/min, and the bundle of His and Purkinje fibers at 20-30 beats/min. In normal conditions, the action potential is initiated in the S-A node which is located in the right atrium. As its rhythm is faster than the rest pacemaker regions, it causes them to excite, before they get to depolarize themselves. Therefore, the heartbeat is controlled by the S-A node. If this node fails somehow, the A-V node takes control of the heart rate but with a rhythm close to 40 beats/min [2].
The action potential follows the specific path (Figure 2.3 [7] and 2.4):

1. S-A node initiates the action potential
2. Spreads across atria through the internodal pathway towards the A-V node, and via the Interatrial node to the left atrium (takes about 30 ms ensuring unified atrial contraction)
3. Delays for about 100 ms at the A-V node, ensuring the full contraction of the atria and following blood ejection into the ventricles, before the latter start contracting
4. Moves from the A-V node through the bundle of His and into the Purkinje fibers, finally depolarizing the ventricles simultaneously (takes approximately 30 ms)

The path of the action potential through the heart can be recorded externally as an electrocardiogram (ECG). Figure 2.4 shows an electrocardiogram under normal conditions, providing the path of the action potential as well.
The main pattern of the electrical activity was first discovered over a hundred years ago [8]. It contains three waves, named P, QRS (a wave complex), and T.

The P wave is a small deflection wave, that demonstrates the atrial depolarization and the beginning of atrial contraction. The QRS wave complex represents the ventricular depolarization and triggers ventricular contraction. It consists of three waves, Q, R and S. Q wave corresponds to the interventricular septum depolarization, R wave represents the depolarization of the main mass of the ventricles, being reasonably the largest wave, and finally the S wave reflects the final depolarization of the ventricles. Eventually, the T wave signifies the ventricular repolarization triggering ventricle relaxation.

2.1.1.4 The Cardiac Cycle

The cardiac cycle contains a time period of relaxation which is called diastole, during which the heart fills with blood (isovolumic relaxation and diastolic filling phases). This phase is followed by a period of contraction called systole (isovolumic contraction and ejection phases), during which the blood gets ejected to the different parts of the body. In normal conditions the cardiac cycle lasts about 800 ms, from which, the 500 ms refer to the diastolic phase and the rest 300 ms represent the systolic phase. Figure 2.5 [4] illustrates the different events taking place in the left part of the heart.

![Figure 2.5. Events of the cardiac cycle for ventricular function – Wiggers Diagram [4]](image)

Department of Mechanical Engineering & Aeronautics - Division of Energy, Aeronautics & Environment
In Figure 2.6, left atrial, left ventricular and aortic pressure is depicted. Ventricular volume changes, the ECG and the phonocardiogram are also presented.

During diastolic filling, the atria and ventricles are both relaxed. The left and right atria receive blood from the venous system (pulmonary veins for the left, vena cava for the right atrium), resulting in a pressure value which keeps the A-V valves open. During this phase no contraction of the atrium walls takes place, and the blood flows unhindered towards the ventricles. It is when the P wave and the depolarization of the atria occurs that the atrium wall starts contracting. At that time, we have the final ejection of blood into the ventricles. About 0.16 s after the onset of the P wave, the QRS waves appear as a result of the ventricular depolarization and the ventricular contraction starts, initiating the ventricular systole. The A-V valves close after the ventricular pressure exceeds the atrial pressure, and the isovolumic contraction period starts. At this period, blood doesn’t move as all valves are closed, but it doesn’t last long as the ventricular pressure is increasing and finally exceeding the pressure in aorta and pulmonary artery, causing the opening of the semilunar valves. Pressure continues to increase, until the contraction ends, despite the ventricular volume decrease. Finally, the T wave initiates ventricular relaxation, and the ventricular pressure starts decreasing, leading to the closure of the semilunar valves. After this, ventricular diastole and isovolumic relaxation arise. The A-V valves remain closed until the atrial pressure becomes higher than ventricular. This sums up the events happening during one cardiac cycle [2, 4].

What’s really interesting about the atria is that the contraction is only responsible for the 20 % of the blood volume that fills the ventricles, meaning that at least in resting conditions, when the atria fail to function, the difference is unlikely to be noticed. Apart from that, in Figure 2.5, we can notice some pressure changes in the atrial pressure curve. These elevations are called a, c, and v atrial pressure waves. The a wave is a result of the atrial contraction, and the c wave occurs at the start of ventricular contraction. The v wave shows up in the end of ventricular contraction [4]. The right atrial pressure is dominated by the a waves, achieving a mean pressure normally less than 5 mmHg, whereas in the left atrial pressure waveform the v wave is dominant [9]. The “x” and “y” depicted in Figures 2.6 and 2.7 are just the descents occurring [10].
2.1.2 Blood Vessels

As mentioned, blood reaches every part of the body from the heart through the blood vessels. There are three types of blood vessels: the arteries, the veins, and the capillaries.

The arteries act as distributors of blood from the heart to the tissues. The majority of them, apart from the pulmonary and umbilical arteries, carry oxygenated blood from the left ventricle to other organs and tissues. As their distance from the heart increases, they branch out and form smaller arteries, which lead to even smaller vessels, the arterioles, that eventually end in capillaries. The capillaries have very thin walls so they can allow the exchange of substances between blood and cells. They have no valves and connect the arteries to the veins (venules). Finally, the veins return the blood from the tissues back to the right atrium of the heart. At any given time, the veins contain 75% of the amount of blood, while the arteries 20% and capillaries the rest. Thus, one can assume that the veins function as reservoirs of blood. Figure 2.8 shows the distribution of blood volume among the different types of blood vessels in the circulatory system [4].

![Image of blood vessels](image.png)

*Figure 2.8. Blood volume distribution among the vessels [4]*

<table>
<thead>
<tr>
<th>Vessel</th>
<th>Diameter (mm)</th>
<th>Length (mm)</th>
<th>Wall thickness (μm)</th>
<th>Pressure (mmHg)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aorta</td>
<td>25</td>
<td>400</td>
<td>1500</td>
<td>100</td>
</tr>
<tr>
<td>Large Arteries</td>
<td>6.5</td>
<td>200</td>
<td>1000</td>
<td>100</td>
</tr>
<tr>
<td>Arterioles</td>
<td>0.1</td>
<td>2</td>
<td>20</td>
<td>60</td>
</tr>
<tr>
<td>Capillaries</td>
<td>0.008</td>
<td>1</td>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>Venules</td>
<td>0.15</td>
<td>2</td>
<td>2</td>
<td>20</td>
</tr>
<tr>
<td>Large Veins</td>
<td>14</td>
<td>200</td>
<td>800</td>
<td>10</td>
</tr>
<tr>
<td>Vena Cava</td>
<td>30</td>
<td>400</td>
<td>1200</td>
<td>5</td>
</tr>
</tbody>
</table>

*Table 2.1. Approximate quantification of individual vessels in the circulatory system*

Table 2.1 [11] which is based on bibliographical data, includes information about typical geometrical characteristics and pressure values of different blood vessels. Data refer to a 30-year-old male, with mass 70 kg and 5.4 L blood volume. In Figure 2.9 [4], pressure distribution among
The cardiovascular system

the systemic and pulmonary circulation is presented. It is worth mentioning that, in the systemic circulation due to the pulsatile nature of blood flow, we receive the depicted fluctuations in the pressure distribution. These fluctuations actually refer to the systolic (120 mmHg) and diastolic (80 mmHg) pressure levels. Apart from that, it can be noticed that the mean pressure falls progressively to approximately 0 mmHg until blood reaches the smaller veins and finally venae cavae during the systemic circulation. On the other hand, in the pulmonary circulation, as the resistance to the flow is lower, the pressure values are respectively lower with a maximum systolic pressure near 25 mmHg and a diastolic pressure close to 7 mmHg (in the pulmonary arteries).

For the completeness of this thesis, it is important to make a reference to the mean blood pressure. As previously mentioned, we can detect the systolic and diastolic pressure values, 120 and 80 mmHg respectively (normally). However, the mean pressure (Mean Arterial Pressure – MAP) is less than the arithmetic average of the systolic and diastolic pressure, as systole lasts for about one third of a cycle while diastole, for about two thirds [11, 12]. Therefore, the mean blood pressure (MAP or $P_{\text{mean}}$) is a weighted sum, as described in Eq.(2.3),

$$P_{\text{mean}} = \frac{P_{\text{systole}} + 2P_{\text{diastole}}}{3},$$

where, $P_{\text{systole}}$ is the systolic and $P_{\text{diastole}}$ the diastolic pressure.

For normal values of 120 mmHg for $P_{\text{systole}}$ and 80 mmHg for $P_{\text{diastole}}$, we gain a mean value of around 93 mmHg. In addition to MAP, we can define pulse pressure as well. Arterial Pulse Pressure is the difference between systolic and diastolic pressures in the aortic (Figure 2.10). For instance, if the systolic pressure is 130 mmHg and the diastolic pressure is 85 mmHg, then the pulse pressure is 45 mmHg.

![Figure 2.9. Pressure fluctuations in different parts of the cardiovascular system [4]](image1)

![Figure 2.10. MAP and Pulse Pressure [11]](image2)
2.1.3 Blood

Blood is a non-Newtonian complex fluid, due to its inhomogeneous composition. It consists of plasma, erythrocytes (known as red blood cells, 45% of the total blood volume), leukocytes (white blood cells, 0.3%), and platelets (0.15%). The red blood cell volume fraction is called hematocrit. One of the reasons that a person’s hematocrit is clinically relevant is that an increase in hematocrit contributes to a dramatic increase in blood viscosity [14] (Figure 2.11 [12]).

The red blood cells have the shape of a toroidal disc, 7.5 μm in diameter and a maximum thickness of 2 μm. Their diameter is approximately equal to the inner diameter of the capillaries, however because of their deformability, they can even fit in even smaller diameters. The leukocytes are spherical with a diameter of 7 μm, while platelets are even smaller. The blood plasma is a newtonian fluid, consisting mostly of water (90%) and has a dynamic viscosity of about 0.0012 Pa-s (at 37 degrees Celsius) [12], [13].

For very slow shear rates (definition of shear rate will be discussed later), blood viscosity is 100 times higher than water viscosity, while for high shear rates, condition which is detected in larger vessels, this is about 4 times higher than water, with a value of 0.004-0.005 Pa-s [12].

![Figure 2.11. Blood viscosity vs. hematocrit. Sharp increase observed in blood viscosity after HCT>40 [12]](image)

2.2 The Left Atrium

In the present subchapter we will focus on the left atrium morphology and function. The left atrium (LA) is not just a simple chamber used for the transportation of blood, but it is highly dynamic and of great clinical importance [15]. Left atrial enlargement has been linked with risk of stroke, heart failure, development of atrial fibrillation, and even risk of death [16].

In fluid mechanics terms, the left atrium geometry consists of 4 inlets (most of the times) and one outlet. As can be observed in Figure 2.12 [19] there is an inflow of oxygenated blood into the atrium through the four pulmonary veins RSPV, RIPV, LSPV, LIPV (R stands for right, L for left, S for superior and I for inferior). The blood exits the left atrium via the mitral valve, which has
been described previously. A remarkable area of medical interest is the left atrial appendage (LAA). LAA is a long, tubular, hooked structure derived from the left atrium [18], that plays a critical role in thrombus formation as a result of atrial fibrillation (see Subchapter 2.3).

**Posterior view**

![Posterior view diagram](image)

**Lateral view**

![Lateral view diagram](image)

After a quick look into the left atrial geometry, we can now examine its function, which has been divided into three phases [15]. First, the left atrium functions as a reservoir during the LV contraction and isovolumetric relaxation, which means that blood coming from the pulmonary veins stays in the atrium as the mitral valve is closed. Secondly, the LA acts a conduit, meaning that blood is transferred passively from LA to LV. Finally, the LA actively contracts while the final phase of diastole takes place, functioning as a pump. Figure 2.13 [20] illustrates these 3 phases in a diagram of LA volume versus LA pressure.

**Figure 2.12. Different locations of Left Atrial Geometry [19]**

<table>
<thead>
<tr>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Roof - right</td>
</tr>
<tr>
<td>2. Roof - middle</td>
</tr>
<tr>
<td>3. Roof - left</td>
</tr>
<tr>
<td>4. Posterior wall - right</td>
</tr>
<tr>
<td>5. Posterior wall - middle superior</td>
</tr>
<tr>
<td>6. Posterior wall - middle</td>
</tr>
<tr>
<td>7. Posterior wall - left</td>
</tr>
<tr>
<td>8. 'Floor' - right</td>
</tr>
<tr>
<td>9. 'Floor' - middle</td>
</tr>
<tr>
<td>10. 'Floor' - left</td>
</tr>
<tr>
<td>11. Mitral isthmus origin</td>
</tr>
<tr>
<td>12. Left lateral ridge</td>
</tr>
</tbody>
</table>

**Figure 2.13. Left atrial pressure-volume loop demonstrating left atrial reservoir, conduit, and pump functions. MV/O-C refers to opening and closure of MV [20]**

Department of Mechanical Engineering & Aeronautics - Division of Energy, Aeronautics & Environment
The left atrial volume is mostly characterized by the left atrial volume index (LAVI). The normal range for LAVI values is between 21 and 52 mL/m² [21]. Moreover, the left atrial end-systolic volume index (LAESVI) can also act as a predictor of cardiovascular outcomes, and it is recommended for the measurement of the left atrial size [22].

2.3 Atrial Fibrillation

2.3.1 Definition of Cardiac Arrhythmias

Atrial Fibrillation (also known as AF-AFib) is the most common cardiac arrhythmia, that affects 1% to 2% of the general population [23]. Arrhythmias are abnormalities in the heartbeat meaning that the heart could beat too quickly, too slowly, or with an irregular rhythm. Under normal conditions the heart is said to be under sinus rhythm, as the sinoatrial node (SA) is fully controlling the cardiac rhythm. A normal sinus rhythm ranges from 60 to 100 beats/min [11]. A sinus rate less than 60 beats/min is concerned as a sinus bradycardia, whereas a sinus rate of 100-180 beats/min is called sinus tachycardia. These 2 conditions are relatively normal, and they take place in different occasions, like resting and exercising respectively.

Arrhythmias are abnormalities of the heart’s electrical properties. These disorders refer to either an abnormality in the generation of an electrical impulse (abnormality in automaticity) or an abnormality in impulse conduction after the generation of the impulse (abnormality in propagation). There can be also a distinction between bradyarrhythmias and tachyrrhythmias. Finally, a classification of the arrhythmias can be made as a result of the anatomical site of origin, for example, atrial, atrioventricular junctional or ventricular (e.g. ventricular fibrillation)[24].

2.3.2 Atrial Fibrillation: definition, causes and management

Atrial Fibrillation is the most frequent sustained arrhythmia worldwide, with significant morbidity and mortality rates, especially in the elderly. AF is identified by rapid (more than 400 beats/min) and disorganized atrial activation, resulting in an impaired atrial functionality, which can be diagnosed by an ECG, due to the lack of the P wave and the irregular QRS complexes [24].

![Figure 2.14. Atrial Fibrillation vs normal sinus rhythm. Disorganization of the signals in atria during AF [18 ]](image)
Cells in different regions in the atria depolarize, repolarize and are excited again randomly [14]. Figure 2.14 illustrates the differences that can be noticed through an ECG between normal and AF conditions. Atrial fibrillation can be classified in three types, based on duration. AF terminating within 7 days is called paroxysmal, whereas AF that is present for more than 7 days is named persistent AF (Long-standing persistent accounts for continuous AF for at least a year). AF that does not spontaneously convert to normal sinus rhythm is classified as permanent [1].

Due to the disorganized atrial activity, the pumping ability of the left atrium is decreased or even not occurring at all. Therefore, it becomes useless as primer pump for the ventricle. Yet, as previously mentioned, the atrium is only responsible for the 20% of the ventricle filling, meaning that a person can live for months or even years under the condition of AF, despite the reduced total pumping efficiency [4].

The mechanisms behind it are not well understood, but impulses are thought to progress repeatedly around abnormal conduction paths, which sometimes are called circus pathways, based on a re-entry phenomenon. The atrial conduction system is altered, in such a way that a single wave of excitation does not die out, yet continues to travel around an altered conduction loop in the atrium. As a result, this continuous activity may drive the atria and the atrioventricular node at a very high frequency. The described process is known as a reentry phenomenon, and it’s depicted in Figure 2.15 [14].

![Figure 2.15. Normal and reentrant excitation pathways [14]](image)

The current theory implies two distinct, though equally important, pathophysiological components; the presence of multiple rapidly firing ectopic foci located mostly around the pulmonary veins, that trigger initiation of the arrhythmia and the existence of an abnormal tissue substrate that is capable of maintaining the arrhythmia [25].

Ectopic foci are abnormal pacemakers sites in the heart (not in the SA node) that displace automaticity, the property of cardiac cells to generate spontaneous action potentials [26]. They’re found to be located within the structure of the pulmonary veins close to the left atrium (also found in the right atrium, Figure 2.16 [27]).
Apart from the triggers, in order for the atrial fibrillation to keep going, a tissue substrate capable of maintaining AF is required. The reentry phenomenon within the atrial myocardium is observed and is facilitated by conduction slowing and shortening of the refractory period [25]. The time period after the initiation of an action potential, during which the cardiac cell is incapable of initiating another action potential, is called refractory period.

Risk factors possibly responsible for the initiation of atrial fibrillation include hypertension, heart failure, diabetes, peripheral vascular disease, ageing and lifestyle. Left atrial dilatation (enlargement) is also believed to contribute as well [28, 29].

The clinical outcomes of atrial fibrillation contain stroke events, extracranial systemic thromboembolism, dementia, heart failure, myocardial infraction, and eventually mortality. In this thesis, a reference will be made only for stroke, as it is the most common thromboembolic event associated with AF and dramatically increases the risk of long-term disability, or death [30], [29]. The genesis of thrombus hasn’t been fully understood, yet a participation of three factors, blood stasis, endothelial dysfunction and prothrombotic state, has been proposed.

Attention has been devoted to the Left Atrial Appendage (LAA – Figure 2.12). Loss of contractility of the left atrium, and especially in the LAA, due to AF, can lead to local stasis (very low blood velocity) and thrombus formation [31]. This thrombus might finally end up in the systemic circulation and cause a stroke event. Therefore, it is reasonable to think that the elimination of the LAA might act as a preventive strategy for AF-related stroke.

There is a variety of prevention and treatment options based on the severity of the disease of each patient. There are both pharmaceutic and non-pharmaceutic solutions offered. Pharmaceutic solutions include anticoagulant drugs that reduce the risk of stroke events, antiarrhythmic drugs (rhythm control) and drugs for reducing the ventricular response rates [32].
Surgical options are kind of minimally invasive surgeries, such as the catheter ablation procedure and the left atrial appendage closure. Catheter ablation has been a well-established treatment option for patients with symptomatic atrial fibrillation. Despite the fact that catheter ablation is a safe procedure, its usage remains mostly as a second-line therapy option, after pharmaceutical therapy has failed [33]. Catheter ablation is actually the delivery of energy through a catheter to create a lesion eliminating a pathway or a structure that is responsible for the initiation or the maintenance of the arrhythmia and for our case, AFib. The kind of energy that has been used mostly is the radiofrequency. A more recent technology, named cryoablation is also used and it’s responsible for over a third of the AFib ablations done [30]. Figure 2.16 [35] shows schematically the ablation procedure.

Figure 2.17. Catheter ablation (cryoablation and radiofrequency current) procedure for the isolation of the pulmonary veins [35]

The ablation of AF is a procedure of isolating the pulmonary veins from the rest of the LA, as previously described, this is where the triggering points (the ectopic foci) are located. Except for the pulmonary veins, some other targets, less frequently ablated, are regions with low voltage, that are possibly responsible for the maintenance of AF [34]. Something that has not been mentioned yet, is that before or during the procedure of catheter ablation an electrophysiological (voltage and anatomical) mapping of the left atrium is carried out, during which, the doctors decide which regions they are going to ablate.

Figure 2.18 [33] depicts an electrophysiological mapping of the left atrium after pulmonary vein isolation. The spheres represent the points of ablation. Red color indicates voltage values under 0.1 mV and purple over 0.5 mV. After the procedure, it is mentioned that the patient’s ejection
fraction improved from 25% to 55%. It is of paramount importance to mention that the main aim of this thesis is to create maps of Wall Shear Stress indices that will be then compared to the electrophysiological maps, provided by the doctors participating in this project, in order to examine whether any connection between the electrical (e.g., Voltage) and mechanical (e.g., WSS) factors mentioned can be found. This possible correlation might provide doctors with robust data in the direction of a more efficient ablation procedure.

Finally, another option for those who are not good long-term candidates for anticoagulation drugs and are at an increased risk for a stroke event, is the occlusion (closure) of the left atrial appendage (LAA). In nonvalvular AF 90% of clots are found in the LAA [36]. Several devices have been proposed for this reason, including solutions achieved from an outside perspective or an inside (endovascular). The most known one is presented in Figure 2.19 [37].
3 Fluid Mechanics & Hemodynamics

The flow of blood in the cardiovascular system, within the heart and in the arteries for example, is based on the same principles as the flow around an aircraft wing or the flow of oil in a tube. These fundamental principles apply to fluid mechanics. In general, fluid mechanics is the science that studies the fluid’s behavior (liquids, gases, and plasmas) at rest and in motion. Fluid mechanics has a wide spectrum of applications, yet, in the present thesis only the basic principles will be given, including applications in hemodynamics.

3.1 Fluid Mechanics

3.1.1 Properties of fluids

The fluids themselves are divided in the following three categories:

- Ideal fluids are called the non-real fluids, which are not compressible and do not include internal friction between their molecules during their flow, that is they have zero cohesion (viscosity). Needless to say, that they’re imaginary, and they are used just in order to help researchers make simplified calculations and draw general conclusions.

- Newtonian fluids are fluids that satisfy the Newton’s experimental law. For that case, shear stress, \( \tau \) (Eq. (3.1)), is linearly related to the rate of shearing strain (also referred to as rate of angular deformation [38]).

\[
\tau = \mu \frac{\partial u}{\partial y} \quad (3.1)
\]

In Eq. (3.1), \( \mu \) is the dynamic viscosity of the fluid (Pa·s or kg/m·s) and \( \frac{\partial u}{\partial y} \) refers to the rate of shearing strain (also written as \( \dot{\gamma} \) (1/s))

- Non-Newtonian fluids, in contrast, are fluids for which the shearing stress is not linearly related to the rate of shearing strain.

Dynamic viscosity \( \mu \) is related to kinematic viscosity \( \nu \) (m²/s), as described in Eq. (3.2),

\[
\nu = \frac{\mu}{\rho} \quad (3.2)
\]

where \( \rho \), is the density of the fluid (kg/m³).

3.1.2 Laminar and Turbulent Flow

In this part of the thesis, different patterns of the flow will be discussed, mostly regarding flows inside a pipe that resemble the flows in blood vessels.

Laminar flow is the type of a flow pattern, that has a stratified character, where the flow rates fluctuate with the change of flow status, but do not present local or temporal fluctuations. The opposite of laminar flow is turbulent flow. Turbulence deriving from the Greek word “\( \tau\upbeta\beta\eta \)” means noise, disorder and chaos. Turbulent flow is characterized by intense, irregular and even chaotic fluctuations of flow magnitudes, such as pressure and velocity, yet, those fluctuations are not random, as the show some kind of structure. In practical applications, turbulent flows are the rule, while laminar is the exception. For instance, heart sounds that are detected by the stethoscope are product of turbulence (e.g. valves closure). In abnormal conditions, the presence of
Atherosclerosis contributes to the occurrence of turbulence, leading to abnormal sounds that can be detected by the doctor (e.g. atherosclerosis in carotid artery).

A comparison of the mentioned flow patterns is illustrated in Figure 3.1 [39]. We can clearly observe the flow particles moving in a single layer for the laminar flow conditions, whereas in the turbulent, particles move erratically between layers. The transition from laminar to turbulent flow does not occur suddenly. This happens in a region where the flow fluctuates between laminar and turbulent flows before it becomes fully turbulent.

This transition depends on several factors, including geometry, surface roughness, flow velocity, surface temperature and type of fluid [40].

It was discovered by Osborne Reynolds that the flow regime is mostly related to the ratio of inertial forces to viscous forces within the fluid. This ratio is known as Reynolds number (Re) and for the case of a flow inside a pipe (these flows are considered internal flows, meanwhile the flow around an airfoil is considered external), it is expressed as described by Eq (3.3),

$$ Re = \frac{\text{Inertial forces}}{\text{Viscous forces}} = \frac{V_{avg}D}{\nu} = \frac{\rho V_{avg}D}{\mu}, $$

where $V_{avg}$ is the average flow velocity (m/s), $D$ is the characteristic length of the geometry (for a pipe flow, the diameter (m)), and $\nu$ and $\mu$ the kinematic and dynamic viscosity respectively.

The critical Reynolds number at which the flow becomes turbulent (Re$_{ct}$), for internal flows in circular pipes is accepted to be 2300. In general, we consider:

- $Re \leq 2300$ as laminar flow
- $2300 \leq Re \leq 4000$ transitional flow
- $Re \geq 4000$ turbulent flow

Figure 3.2 [41] shows the fully developed velocity profiles of a pipe flow. Laminar flow (a) presents a parabolic profile, whereas in turbulent flow (b) a fairly flat velocity distribution is observed across the pipe’s section. Finally, the transition to turbulent flow leads to higher pressure losses, as turbulent is energetically more costly than laminar flow [42].
3.1.3 Entrance Region and Entry Lengths

Definitely, any fluid flowing inside a pipe entered at some point of time the pipe. The region of flow close to where the fluid enters the pipe is called entrance region. This region is important because its characteristics define whether the flow is fully developed or not.

As the fluid enters the pipe, the fluid particles in the layers near the walls of the pipe come to complete stop as the walls have zero velocity and they don’t move (no slip boundary condition). These layers also slow down the above to them layers due to the friction between them. The region of the flow where the effects of viscous shearing forces are observed is known as a boundary layer. The thickness of the boundary layer increases across the length of the pipe, until it reaches the pipe center. The region from the pipe inlet to the point where the boundary layer reaches the pipe center is called entrance region and the length between them, hydrodynamic entry length $L_h$. Finally, the region after which the velocity profile remains unchanged is called hydrodynamically fully developed region (Figure 3.3 [40]).

For laminar flow, the hydrodynamic entry length is given approximately as:

$$L_h = \frac{8\nu}{D_v}$$
where, Re is the Reynolds number and D, the diameter of the pipe.

The entry length in turbulent flows is less depended on Reynold number. It is usually approximated as described in Eq.(3.5).

\[ L_{h,\text{turbulent}} \approx 10D \]  

### 3.1.4 Governing equations of fluid motion

The flow of fluids is described by 3 sets of equations: the continuity equation (conservation of mass), the Navier-Stokes equations (Newton’s second law or momentum equation) and energy conservation equation. In contrast to one-dimension equations, 3D equations of the flow that describe real conditions, are much more complex. We consider a cartesian system of coordinates with velocities u, v, w in all three dimensions (x,y,z) as followed:

\[ u = u(x, y, z, t), \quad v = v(x, y, z, t), \quad w = w(x, y, z, t), \]  

where the vector velocity field \( \vec{V} \) is shown in Eq.(3.7);

\[ \vec{V} = u \vec{i} + v \vec{j} + w \vec{k} , \]  

where i, j, k are the unit vectors along the x, y, z axes.

#### 3.1.4.1 The continuity equation

The continuity equation which describes the conservation of mass is given by Eq.(3.8).

\[ \frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} + \frac{\partial (\rho v)}{\partial y} + \frac{\partial (\rho w)}{\partial z} = \frac{Dp}{Dt} + \rho \vec{\nabla} \vec{V} = 0 \]  

For steady flows, changes in time do not occur, therefore \( \partial / \partial t = 0 \). At the same time for incompressible flows, density \( \rho \) remains unchanged in space. Hence, for incompressible and steady flows, Eq.(3.9) is simplified as presented in Eq.(3.9).

\[ \nabla \vec{W} = div \vec{V} = 0, \]  

with div, referring to divergence of velocity.

#### 3.1.4.2 The momentum equation

The physical principle behind the following set of equations is Newton’s second law (F=ma). We consider the balance of forces acting on infinitiesmally small fluid element as shown in Figure 3.4 [43]. The forces acting, are either body forces acting directly on the volumetric mass and surface forces, which act directly on the surface of the fluid element.

**Figure 3.4. Infinitesimally small fluid element and forces in the x direction only [43]**
On each side of this finite volume $dV = dx dy dz$, shear ($\tau$) and normal stresses ($\sigma$) act, which are balanced, according to Newton’s law, with the inertia and field forces such as gravity and magnetic forces. Equation (2.10a,b,c) applies to shear stresses.

$$\tau_{xy} = \tau_{yx} = \mu \left( \frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right) \quad \tau_{yz} = \tau_{zy} = \mu \left( \frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right) \quad \tau_{zx} = \tau_{xz} = \mu \left( \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \tag{3.10a,b,c}$$

For isotropic newtonian fluids, normal stresses are given (if we exclude pressure) by Eq. (3.11a,b,c).

$$\sigma_x = -\frac{2}{3} \text{div}\vec{V} + 2\mu \frac{\partial u}{\partial x} \quad \sigma_y = -\frac{2}{3} \text{div}\vec{V} + 2\mu \frac{\partial v}{\partial y} \quad \sigma_z = -\frac{2}{3} \text{div}\vec{V} + 2\mu \frac{\partial w}{\partial z} \tag{3.11a,b,c}$$

The shear forces that include both normal and shear stress, are presented in Eq(2.12a,b,c).

$$T_x = \frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau_{yx}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} \quad T_y = \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \sigma_y}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} \quad T_z = \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial \sigma_z}{\partial z} \tag{3.12a,b,c}$$

The connection of the friction forces with the inertia forces and with the field forces (X,Y,Z) is made by the Euler equation which is given for the direction x in Eq.(3.13).

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + X \tag{3.13}$$

We can now receive the general form of the momentum equations by adding the shear force for all the dimensions, as shown in Eq (2.14).

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \vec{X} + T_x \quad \rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \vec{Y} + T_y \quad \rho \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \vec{Z} + T_z \tag{3.14a,b,c}$$

Finally, for the general case of flow, that is unsteady, compressible, viscous and three dimensional we obtain the following equations for newtonian fluids known as Navier-Stokes equations.

$$\rho \frac{Du}{Dt} = -\frac{\partial p}{\partial x} + \vec{X} + \frac{\partial}{\partial x} \left( \mu \left( 2 \frac{\partial u}{\partial x} - \frac{2}{3} \text{div}\vec{V} \right) + \frac{\partial}{\partial y} \left( \mu \left( \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \right) + \frac{\partial}{\partial z} \left( \mu \left( \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \right) \tag{3.15.a}$$

$$\rho \frac{Dv}{Dt} = -\frac{\partial p}{\partial y} + \vec{Y} + \frac{\partial}{\partial y} \left( \mu \left( 2 \frac{\partial v}{\partial y} - \frac{2}{3} \text{div}\vec{V} \right) \right) + \frac{\partial}{\partial z} \left( \mu \left( \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \right) \tag{3.15.b}$$

$$\rho \frac{Dw}{Dt} = -\frac{\partial p}{\partial z} + \vec{Z} + \frac{\partial}{\partial z} \left( \mu \left( 2 \frac{\partial w}{\partial z} - \frac{2}{3} \text{div}\vec{V} \right) \right) + \frac{\partial}{\partial x} \left( \mu \left( \frac{\partial w}{\partial x} + \frac{\partial u}{\partial z} \right) \right) + \frac{\partial}{\partial y} \left( \mu \left( \frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right) \right) \tag{3.15.c}$$

In general, we combine the three components into one vector equation and considering constant viscosity we obtain the following Incompressible Navier-Stokes equation:

$$\rho \frac{D\vec{V}}{Dt} = -\nabla p + \rho \vec{g} + \mu \nabla^2 \vec{V} \tag{3.16}$$
For pipe related flows, the Navier-Stokes equations can be expanded in cylindrical coordinates $(r, \theta, z)$ and $(u_r, u_\theta, u_z)$.

### 3.1.4.3 Energy Equation

The energy equation depends on the physical principle of energy conservation. Without any further explanation it is given in its general form in Eq.(3.17):

$$\frac{DE}{Dt} = \rho \frac{De}{Dt} + \rho \frac{D}{Dt} (u^2 + v^2 + w^2),$$  \hspace{1cm} (3.17)

where $e$, is the internal energy per unit of mass.

### 3.1.5 Exact solutions to Navier-Stokes Equations

It’s needless to say that it’s impossible to solve every problem using the exact same Navier-Stokes equations, as the complexity of the real-world problems makes it extremely hard. However, for simple problems like the laminar flow between two parallel plates or Couette flow, exact solutions have been proposed. An interesting solution to a problem that applies to the present work is the laminar axisymmetric flow in a pipe, which is known as Hagen-Poiseuille flow, as it can simulate the flow in a cylindrical blood vessel.

![Figure 3.5. Hagen-Poiseuille Flow [42]](image)

For the solution of the Navier-Stokes equation, cylindrical coordinates are used, while zero rotational velocity is implemented. As it can be observed in Figure 3.5 [42], a parabolic velocity profile is obtained, which is also confirmed by Eq.(3.18), a solution to N-S equations.

$$u(r) = \frac{\Delta p}{4\mu} (r^2 - r_i^2),$$  \hspace{1cm} (3.18)

where, $r_i$ is the radius of the cylinder, $r$ is the radius starting from the center of the cylinder section, $\mu$ the dynamic viscosity and $\Delta P/l$ is the pressure loss over the tube of length $l$.

Velocity is maximal at the axis ($r = 0$) (Eq.3.19), and minimal (zero) at the walls. Mean velocity proved to be half the maximal velocity and it is found to be at $r \approx 0.7r_i$. 
Fluid Mechanics & Hemodynamics

Finally, the pressure loss between two sections (diameter-d) of the pipe located at a distance l is given by Eq.(3.21).

\[
\Delta p = p_1 - p_2 = \frac{64}{Re} \frac{l}{d} \rho \overline{u}^2
\]  

(3.21)

3.1.6 A note on turbulent flows

The flow in the cardiovascular system is not always laminar, but in some conditions (e.g. exercise) and in some parts of it (heart chambers and big blood vessels), it becomes turbulent. As previously said, the fundamental difference between laminar and turbulent flow lies in the chaotic and random behavior of the various flow magnitudes. One of these parameters, is the velocity field, which under turbulence conditions presents fluctuations as shown in Figure 3.6 [38].

As presented in Figure 3.6, such flows and fields can be evaluated by their mean values (\( \overline{u} \)) which take into account the fluctuations in the time field. For a three-dimensional field the mean value of velocity is calculated as described by Eq.(3.22):

\[
\overline{u} = \frac{1}{T} \int_{t_0}^{t_0+T} u(x, y, z, t) \, dt,
\]  

(3.22)

where T, is the time interval.

The fluctuating part of velocity, \( u' \) is that time-varying portion that differs from the average value. We can assume then:
Taking into consideration the above, someone could assume that the viscous shear stress (Eq. (3.1) for laminar flow) could be derived in the same way using the mean value of velocity $\bar{u}$. Nevertheless, it is found that $\tau \neq \mu \frac{du}{dy}$, as many experimental and theoretical studies showed that this approach leads to false results [38]. A lot of concepts of turbulent flow should be taken into account such as the turbulence intensity and the eddy viscosity, but some of them will be discussed in the next chapter concerning the simulation of turbulent flow.

### 3.1.7 Vortex Dynamics

As it will be mentioned later, vortex structures are generated within the heart chambers, and are of special significance when it comes to the left atrium under atrial fibrillation conditions.

Vorticity is a fundamental parameter of the flow field, and it represents the local rotation of fluid particles. Vorticity is a vector ($\omega(t,x)$) and it is defined as the curl of the velocity field as shown in Eq. (3.24). Equation (3.25) represents the calculation of vorticity in cartesian coordinates.

$$\omega(t, x) = \nabla \times u$$  \hspace{1cm} (3.24)

$$\begin{bmatrix}
\omega_x \\
\omega_y \\
\omega_z \\
\end{bmatrix} = \begin{bmatrix}
\frac{\partial u_z}{\partial y} - \frac{\partial u_y}{\partial z} \\
\frac{\partial u_x}{\partial z} - \frac{\partial u_z}{\partial x} \\
\frac{\partial u_y}{\partial x} - \frac{\partial u_x}{\partial y}
\end{bmatrix}$$  \hspace{1cm} (3.25)

Vorticity is not limited to local rotation only, but it describes the flow in general, as it is thought to be the “skeleton” of the flow field [63]. We can loosely describe a vortex as a fluid structure with a circular or swirling motion. The intensity of a vortex is a measure of its circulation, $\Gamma$. In addition, vorticity is a field with zero divergence as shown in Eq. (3.26)

$$\nabla \cdot \omega = 0$$  \hspace{1cm} (3.26)

The Navier-Stokes equation can be as well rearranged in terms of vorticity.

$$\frac{\partial \omega}{\partial t} + u \cdot \nabla \omega = \omega \cdot \nabla u + \nu \nabla^2 \omega,$$  \hspace{1cm} (3.27)

where, velocity $u$ can be re-written as a decomposition of the whole velocity field that includes both the rotational component $u_{rot}$ and the irrotational component $u_{irr}$ that is independent from the vorticity.

$$u = u_{rot} + u_{irr}$$  \hspace{1cm} (3.28)

It’s of critical importance to note that vorticity cannot be generated within the incompressible fluid, yet it only develops from the wall due to the viscous adherence between the fluid and the bounding
structure. The “no-slip” condition (zero velocity) at the interface between the fluid and the solid surface is responsible for the generation of vortices.

The formation of a vortex structure can occur due to the separation of a boundary layer as a result of a local deceleration of the flow. During this event an adverse pressure gradient is present, possibly due to a sudden change in the curvature of the geometry wall such as the carotid artery bifurcations or the aneurysm sac. Moreover, it is possible as well that the irregular contraction of the wall, in the case of AFib, will cause the formation of vortices.

However, how can we detect these vortex structures? The answer to this, is by using several vortex identification methods, which can capture the main vortex structures in the flow field. Hunt et al. (1988) [64, 65, 66], proposed a method, which identified vortices of an incompressible flow as connected fluid regions with a positive second invariant, \( Q \), of \( \nabla \mathbf{u} \), with the additional condition that the pressure in the vortex region should be lower than the ambient value. The second invariant \( Q \) is given in Eq. (3.29),

\[
Q = \frac{1}{2} \left( u_{i,i}^2 - u_{i,j} u_{j,i} \right) = -\frac{1}{2} u_{i,j} u_{j,i} = \frac{1}{2} ( \| \Omega \|^2 - \| S \|^2 )
\]

where \( \| S \| = [\text{tr}(SS^t)]^{1/2}, \| \Omega \| = [\text{tr}(\Omega \Omega^t)]^{1/2} \), and \( S \) and \( \Omega \) are the symmetric and antisymmetric components of \( \nabla \mathbf{u} \), as shown in Eq. (3.30) and Eq. (3.31) (the comma denotes differentiation).

\[
S_{i,j} = \frac{1}{2} (u_{i,j} + u_{j,i}) \quad (3.30)
\]

\[
\Omega_{i,j} = \frac{1}{2} (u_{i,j} - u_{j,i}) \quad (3.32)
\]

When \( Q \) gets a positive value, a vortex structure exists. \( Q \) criterion actually defines vortices as areas where the vorticity magnitude is greater than the magnitude of the rate of strain.

Finally, another criterion that is widely used is the lambda-2 (\( \lambda_2 \)) criterion proposed by Jeong and Hussain (1995) [65]. It is assumed that a pressure minimum is not sufficient as a detection criterion. To solve this issue, a decomposition of the velocity gradient tensor \( J \) (Eq. (3.33)) into its symmetric part, the rate of deformation or strain-rate tensor \( S \), and its antisymmetric part, the spin tensor \( \Omega \) is conducted, and eventually the authors consider only the contribution from \( S^2 + \Omega^2 \).

\[
J = \nabla \mathbf{u} = \begin{bmatrix} \partial_x u_x & \partial_y u_x & \partial_z u_x \\ \partial_x u_y & \partial_y u_y & \partial_z u_y \\ \partial_x u_z & \partial_y u_z & \partial_z u_z \end{bmatrix}
\]

\[
S = \frac{J + J^T}{2}, \quad \Omega = \frac{J - J^T}{2} \quad (3.34)
\]

It is then defined, that a vortex is a connected region where \( S^2 + \Omega^2 \) has two negative eigenvalues, and since \( S^2 + \Omega^2 \) is real and symmetric we will get only real eigenvalues. Afterwards, we consider three eigenvalues \( \lambda_1, \lambda_2, \lambda_3 \) such that \( \lambda_1 \geq \lambda_2 \geq \lambda_3 \). A point of the velocity field is part of the...
vortex structure only if at least two of its eigenvalues are negative ($\lambda_2 < 0$). Nevertheless, when several vortices exist, it can be difficult for the method to detect the individual ones [67]. Figure 3.7 [63] illustrates the vortex formation in a patient with severe mitral stenosis.

![Figure 3.7. Transmitral vortex formation in a patient with severe mitral stenosis. Blue and red represent opposite rotational directions [63]](image)

3.2 Hemodynamics and key hemodynamic indices

Haemodynamics is the study of the properties of blood and how it flows. In this part of the thesis some of the key parameters of blood flow in the cardiovascular system, and especially within the heart, will be presented. Several concepts will be presented but only the left atrium related will be computed in the following parts of this work.

3.2.1 Compliance

Compliance is the amount of distention for a given amount of pressure. Because of it, when the heart pumps blood, the blood vessels don’t act as rigid walls, but they expand, and contract as result of the pressure changes and their elastic nature. The relationship between a change in volume ($\Delta V$) and a change in pressure ($\Delta P$) is the compliance (C), as given in Eq. (3.35).

\[
C = \frac{\Delta V}{\Delta P}
\]  

(3.35)

Therefore, we can realize that compliance is how easy a given pressure causes a change in volume [11]. In biological tissues, the relationship between $\Delta V$ and $\Delta P$ is not linear (Figure 3.8 [11]).

![Figure 3.8. Compliance curve for a biological tissue such as artery [11]](image)
The inverse of compliance is called elastance \( E \), as given in Eq. (3.36):

\[
E = \frac{1}{C}
\]  

(3.36)

Elastance is mostly used for the heart, while compliance is mainly used for the description of blood vessels [42]. Elastance is a similar quantity as stiffness. Figure 3.9 presents the differences in elastance between a normal and a hypertrophied heart. It can be observed that for the same pressure level, the hypertrophied heart has a higher elastance, or in other words, it is stiffer.

Concerning their clinical and physiological significance, the compliance of the arteries decreases with age, and therefore elastance rises, leading to what we know as an increase in pulse pressure (Figure 2.10) with age. This increase in systolic pressure (or pulse) results in an extra load on the heart, which, finally, it is possible that it leads to the hypertrophy of the heart, and the increase of the left ventricular elastance respectively.

### 3.2.2 Resistance

Blood flow through an organ can be described as the pressure gradient (\( \Delta P \)) driving the flow divided by the resistance (\( R \)). For a blood vessel, the pressure gradient is the pressure difference among two points of the vessel. In a similar way to Ohm’s law in electricity, we can obtain its hydrodynamic form in which the current \( I \) equals to the flowrate \( Q \) and voltage difference \( \Delta V \) to pressure difference \( \Delta P \), as shown in Eq. (3.37).

\[
Q = \frac{\Delta P}{R}
\]  

(3.37)

Adding to this the flowrate derived from the Hagen-Poiseuille law (Eq. 3.38) we receive the following:

\[
Q = \frac{\Delta P \cdot \pi \cdot r_i^4}{8 \cdot \mu \cdot l}
\]  

(3.38)
However, the blood flow in the body doesn’t necessarily conform with the above relationship as the latter assumes straight and rigid tubes, newtonian nature of fluid, and finally steady and laminar conditions [11]. Despite these assumptions, this relationship widely is used as it describes the influence of the vessel’s radius on the resistance and the flow in general.

3.2.3 Wall Shear Stress

Wall Shear Stress (WSS) is the force per unit area exerted by the wall on the fluid in a direction on the local tangent plane [44]. Nowadays, WSS distribution among different parts of the cardiovascular system receives a great deal of attention, as it has been described as a mechanism for evolution of vascular disease. Numerous examples can be given. One refers to the case of atherosclerosis at specific areas, like the carotid artery bifurcations. Wall shear stress distribution is also used to evaluate the risk of cerebral and aortic aneurysms rupture.

For 2D and 3D flows the WSS value can be derived from Eq.(3.1) and (3.10). For the specific case of a flow in a circular tube based on Poiseuille law, the WSS ($t_w$) is given as shown in Eq.(3.39).

$$t_w = \frac{4\mu Q}{\pi r_i^3} = \frac{2\mu u_{max}}{r_i}$$  \hspace{1cm} (3.39)

Lower values of WSS show regions with low velocity, that possibly increase the risk of thrombi formation, while higher values indicate high velocity conditions that may lead to the rupture of an aneurysm. Oscillating values of WSS also attract the attention as well.

The Wall Shear Stress can either be measured or calculated implicitly. In vitro measurements (meaning that the studies are performed in laboratory conditions) are carried out in order to calculate wall shear rate or shear stress and include methods such as the laser Doppler anemometry and particle image velocimetry [44]. In vivo measurements (studies take place on living organisms) are mostly simple, and they are based on the Hagen-Poiseuille formula (Eq.(3.39)).

![Figure 3.10. Dependence of flowrate $F(Q)$ on vessel radius [11]](image)

--

Department of Mechanical Engineering & Aeronautics - Division of Energy, Aeronautics & Environment
The only requirement for this calculation is the measurement of the flowrate and the radius of the specific vessel. In addition, the measurement of velocity (or flowrate) can be conducted by 3 ways: non-invasively (e.g. ultrasound, phase-contrast magnetic resonance imaging “MRI” [45]), invasively (intravascular (Doppler) ultrasound) and through Computational Fluid Dynamics (as will be discussed later in the present thesis).

**3.2.4 Time Averaged Wall Shear Stress**

Time Averaged Wall Shear Stress (TAWSS) is described as the wall shear stress averaged over the cardiac cycle, allowing a complete consideration of the blood flow in the specific heart chamber or vessel. Regarding blood vessels, TAWSS is proved to be a well-established in indicating plaque development [48], whereas in the case of left atrial flow, low values of TAWSS, actually indicate low flow velocities, that could be responsible for thrombus formation in the left atrial appendage [49]. TAWSS is calculated as shown in Eq.(3.40):

$$TAWSS = \frac{1}{T} \int_0^T |WSS| \, dt,$$

where, T is the time period of one cardiac cycle.

Before advancing, note that for the calculation of TAWSS, WSS values are absolute values.

**3.2.5 Oscillatory Shear Index**

The Oscillatory Shear Index (OSI) shows the change of direction of the Wall Shear Stress vector from a predominant blood flow direction during the cardiac cycle, and it is defined by comparing the wall shear stress mean with its magnitude [49, 50], as presented in Eq.(3.41):

$$OSI = \frac{1}{2} \left( 1 - \frac{\int_0^T WSS \, dt}{\int_0^T |WSS| \, dt} \right)$$

The calculated values range between 0 a constant flow (the direction of WSS vector does not change at all during the cardiac cycle) for, and 0.5 when the flow is completely inverted during the cardiac cycle.
3.2.6 **Endothelial Cell Activation Potential**

Endothelial Cell Activation Potential index (ECAP) indicates the wall regions that are exposed to both high OSI and low TAWSS values. This ratio specifies the degree of ‘thrombogenic susceptibility’ of the wall [51] and it’s calculated as shown in Eq.(3.42).

\[
ECAP = \frac{OSI}{TAWSS} \tag{3.42}
\]

Regions with high ECAP values (high OSI, low TAWSS) are characterized of endothelial susceptibility (e.g. the LAA).

3.2.7 **Relative Residence Time**

Relative Residence Time index combines wall shear stress and OSI in such a way that it captures the residence time of blood particles near the (atrium) wall [52]. RRT is affiliated with the platelet aggregation in the endothelium [49], and it is calculated as described in Eq.(3.43).

\[
RRT = [(1 - 2 \cdot OSI) \cdot TAWSS]^{-1} \tag{3.43}
\]
4 Computational Fluid Dynamics - Hemodynamics

Fluid motion equations (Navier-Stokes) for compressible and incompressible flows have been known for more than a hundred years. Nevertheless, their complexity and non-linear nature, make them impossible to be used directly for the solution of practical problems, such as the flow over an aircraft or the flow within the heart chambers. As mentioned in Chapter 3.1.5, exact solutions to these equations refer only to very simple geometries and boundary conditions. Consequently, for research and industrial purposes, their numerical solution seems to have no alternative. Although numerical methods, have been first introduced by Newton, due to the obvious lack of computing power, their use began in the late 20th century and until today they have been an integral part of engineering studies, from supersonic aircraft to medical device design. Computational fluid dynamics is based upon the discretization of continuous space.

4.1 Space discretization and numerical schemes

Analytical solutions of partial differential equations (e.g. Wave Equation) are expressed in the continuous field of the physical world (or ‘domain’). In contrast, numerical methods provide solutions at only discrete points of our domain, which are called grid points [43]. Grid, also called mesh, can be modeled in either one or up to three dimensions. Three dimensional grids, provide a more realistic simulation, however, when a detailed simulation is not required, we can use either 2D or 1D conditions, that might still detect the physics of our problem, reducing time and computing power consumption exponentially.

For a two-dimensional grid, the distance between two grid points is called $\Delta x$ for the x direction and $\Delta y$ for y respectively. $\Delta x$ and $\Delta y$ can be either equal or not, while the spacing of the grid points in each direction can be both uniform and non-uniform. These are some options each user selects depending on the problem’s nature. Figure 4.1 [43] shows the discrete grid points in a two-dimensional domain.

![Figure 4.1. Discrete grid points [43]](image)

The most common discretization techniques used are, finite differences, finite volumes and finite elements methods. However, in CFD applications only the first two are operated.
4.1.1 Finite Differences Method

Finite Differences method is the oldest discretization method. It is used mainly for the solution of problems with simple geometry up to 2 dimensions. As mentioned previously, the basis of the finite difference method is the construction of a discrete grid, like the one presented in Figure 4.1, the replacement of the continuous derivatives of the equation by finite differences expressions and the rearrangement of the resulting algebraic equation into an algorithm [53]. The equations that describe the several natural phenomena involve Partial Differential Equations (PDEs), which contain spatial and temporal derivatives (e.g. \( dx, dt \)) and they are usually of first or second order. In order to solve them, an approximation in the discrete space is required.

This approximation is done by using the below simple idea:

\[
\frac{\partial u}{\partial x} = \lim_{\Delta x \to 0} \frac{u(x_0 + \Delta x) - u(x_0)}{\Delta x},
\]

(4.1)

where \( u \) is a flow magnitude, such as the velocity, \( \Delta x \) is the space between two grid points, and \( x_0 \) a grid point in x direction.

If we express the above equation using finite differences, we receive the following approximation:

\[
\frac{\partial u}{\partial x} \approx \frac{u_{i+1} - u_i}{\Delta x}
\]

(4.2)

There is also another way to approximate the first derivative, and this is by using the Taylor series expansion. After processing the series, we get:

\[
\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_i}{\Delta x} - \left( \frac{\partial^2 u}{\partial x^2} \right) \frac{\Delta x}{2} - \left( \frac{\partial^3 u}{\partial x^3} \right) \frac{(\Delta x)^2}{6} + \ldots
\]

(4.3)

The part of the equation next to the known expression of the first derivative is called truncation error. Therefore, as also mentioned, Eq.(4.2) is an approximation of the first derivative, while Eq.(4.3) and (4.4) constitute the full expression. In Equation (3.4), \( O(\Delta x) \) is the truncation error, and it shows what we have neglected using. It also indicates how accurate this method is, what we usually call as the order of the method.

\[
\frac{\partial u}{\partial x} = \frac{u_{i+1} - u_i}{\Delta x} + O(\Delta x),
\]

(4.4)

Equation (3.4) is known as the forward differentiation method and involves \( (\Delta x) \) to the first power indicating that it is first-order accurate. Higher order accuracy can be achieved by using different combinations of the Taylor series expansion, as in most applications in CFD, first order accuracy is not sufficient. Equation (3.5) shows two different finite-difference quotients for first partial derivatives.

\[
\frac{\partial u}{\partial x} = \begin{cases} 
\frac{u_i - u_{i-1}}{\Delta x} + O(\Delta x), & \text{Backward Difference} \\
\frac{u_{i+1} - u_{i-1}}{\Delta x^2} + O(\Delta x)^2, & \text{Central Difference}
\end{cases}
\]

(4.5)
Note that, central differences method is second-order accurate.

For the second derivative, we receive the following central differences method.

\[
\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + O(\Delta x)^2 \tag{4.6}
\]

There are several equations that can be solved numerically. Some of them are the advection equation (e.g. wave equation), Laplace equation (or Poisson), and the diffusion equation. Those can be combined and form more complex equations like the diffusion-advection equation. Moreover, even the Navier-Stokes equations can be analyzed into smaller parts, as diffusion and convective terms are involved.

In an effort to indicate the significance of the numerical solution, an example of the diffusion-advection equation will be presented. In Equations (3.7) and (3.8) the expressions of diffusion and advection terms are presented separately.

\[
\frac{\partial u}{\partial t} = a \frac{\partial^2 u}{\partial t^2}, \quad \text{Diffusion} \tag{4.7}
\]
\[
\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = 0, \quad \text{Advection} \tag{4.8}
\]

where \(a\) is the thermal diffusivity and \(c\) the wave propagation velocity.

The combination of the previous equations is known as the diffusion-advection problem (Eq.(4.9)) and requires initial and boundary conditions in order to be solved.

\[
\frac{\partial u}{\partial t} + c \frac{\partial u}{\partial x} = a \frac{\partial^2 u}{\partial t^2} \tag{4.9}
\]

For its numerical solution, we will use the FTCS method, which contains Forward Differences for the time derivative and Central Differences for the spatial derivative as shown below).

\[
\left. \frac{\partial u}{\partial x} \right|_j = \frac{u_{i+1} - u_{i-1}}{2\Delta x} + O(\Delta x)^2, \quad \left. \frac{\partial^2 u}{\partial x^2} \right|_j = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + O(\Delta x)^2 \tag{4.10}
\]
\[
\left. \frac{\partial u}{\partial t} \right|_n = \frac{u^{n+1} - u^n}{\Delta t} + O(\Delta t) \tag{4.11}
\]

Replacing Eq.(4.10) and (4.11) in Eq.(4.9) we receive the following discretized equation:

\[
u_{j}^{n+1} = \left(-\frac{c}{2} + s\right)u_{j+1}^{n} + (1 - 2s)u_{j}^{n} + \left(\frac{c}{2} + s\right)u_{j-1}^{n}, \tag{4.12}
\]

where \(s = \frac{a\Delta t}{\Delta x^2}\) and \(C\) the CFL (Courant-Friedrichs-Lewy) number which is given as \(C = \frac{u\Delta t}{\Delta x}\).

This method is consistent, and it is 1st order accurate in time and 2nd order accurate in space. However, this method is not always stable, meaning that the numerical error under some conditions, will be increasing and the method will not finally converge to a solution. The FTCS
method for the advection-diffusion problem is stable only as long as the condition of Eq.(4.13) is met.

\[ C^2 \leq 2s \leq 1 \]  

We can notice the dependance of stability of our method on the Courant number. Nevertheless, several other methods can be applied, providing different stability conditions as well. One of them is the Upwind method which involves forward differences for the time derivative, backward differences for the convection term \( \frac{\partial u}{\partial x} \bigg|_j \) and central differences for the diffusion term \( \frac{\partial^2 u}{\partial x^2} \bigg|_j \).

In general, numerous combinations can be introduced for the numerical solution of an equation based on the type of the equation (elliptic, parabolic, hyperbolic). All of them include advantages and disadvantages, hence, it’s up to the user which method it will be used. This, however, requires deep understanding of the problem’s physics. Table 4.1 provides some methods used in relation to the type of the partial differential equation.

<table>
<thead>
<tr>
<th>Elliptic (e.g., Laplace eq.)</th>
<th>Parabolic (e.g., Diffusion eq.)</th>
<th>Hyperbolic (e.g., Advection eq.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jacobi</td>
<td>FTCS</td>
<td>Upwind</td>
</tr>
<tr>
<td>Gauss-Seidel</td>
<td>BTCS</td>
<td>FTCS</td>
</tr>
<tr>
<td>S.O.R.</td>
<td>Crank-Nicolson</td>
<td>Lax-Friedrichs</td>
</tr>
<tr>
<td>Steepest Descent</td>
<td>DuFort-Frankel</td>
<td>Lax-Wendroff</td>
</tr>
<tr>
<td>Conjugate gradient</td>
<td>Richardson</td>
<td>MacCormack</td>
</tr>
</tbody>
</table>

Table 4.1: Numerical methods most commonly used for the solution of elliptic, parabolic and hyperbolic PDEs

After a quick look at the finite differences method, we can now advance into the finite volume method, which is mostly used by several CFD software packages.

### 4.1.2 Finite Volume Method

The Finite Volume Method (FVM) is a numerical technique that replaces partial differential equations representing conservation laws over differential volumes with finite volumes (cells) [54]. In a similar way to the finite difference method, the first step of the solution process is to discretize the geometric domain, and thus we receive non-overlapping elements or finite volumes. After this process, we transform the partial differential equations into algebraic equations after integrating them over each finite volume. Afterwards, we can solve the system of algebraic equations, calculating eventually the values of the dependent variable for each of the cells.

In the FVM, some terms of the conservation equation are turned into face fluxes and evaluated at the finite volume faces. The Finite Volume Method is strictly conservative, since the flux entering a given volume is equal to that leaving the adjacent volume. It can be used in any type of grid, being a suitable method for any complex geometry.
There are two main methods that determine the shape and position of the control volume in relation to the grid. The most common is the cell-centered method, in which flow properties, such as the velocity and pressure values, are defined in the centroid of the grid cells (Cell Center Figure 4.2). The second method is called vertex-centered, as the flow magnitudes are defined at the nodes (See Figure 4.2) of the grid. A difference between these two methods is shown in Figure 4.3. The Cell-centered method is mainly used in OpenFOAM®, and Ansys Fluent®, whereas the Vertex-centered method is used in Ansys-CFX®.

As already mentioned, the FVM is used despite the complexity of the geometry, and regardless of whether the grid is structured or unstructured as shown in Figure 4.4. Structured grids remained the staple in numerical simulation for a long time, yet, in the last two decades, unstructured grids became more popular[54].
Nowadays, a grid can have both structured and unstructured features, as these two, can be combined and generate hybrid grids.

In CFD software several types of elements (cells) are used for the solution of each problem. An element in the finite volume mesh is actually a polyhedron in 3D meshes, or a polygon in 2D meshes. The most commonly used elements are depicted in Figure 4.5.

![Element Types](image)

**Figure 4.5. Most widely used element types [54]**

These different types of elements allow the users to generate meshes, regardless of the geometry complexity. They have both advantages and disadvantages, so the user ought to know which type must be selected before he or she advances with the solution of the case. Multiple topics should be taken into account when deciding the type of mesh, such as the required solution accuracy or the density of the mesh (speaking of quantity of cells).

Two characteristics of a mesh that will be discussed later as well, are skewness and aspect ratio of the mesh elements. The skewness of a grid indicates the quality and suitability of the mesh. Large skewness values of a mesh area will possibly lead in less accurate results. Aspect ratio is the ratio of the longest to the shortest side of a single cell. An ideal value of it is 1, while large values may lead as well to interpolation errors with unknown results.

![Aspect Ratio](image)

**Figure 4.6. Aspect ratio and skewness of 2D elements**
4.2 Computational Hemodynamics

Cardiac diseases are still a major cause of ill health and death in our society, therefore advancements in healthcare are of upmost significance. The ageing population and the so-called obesity epidemic that has gradually expanded through the last decades, have increased the incidence of heart disease. In dedication to treat exclusively these diseases, a new market of medical devices based on computational modelling has risen. The latter has found its way simultaneously with the recent developments in computing and flow visualization technology, thus, providing numerous methods, while facilitating the process of the existing ones as well, for the diagnosis of cardiovascular diseases [58]. This has led to an increasing demand for Computational Hemodynamic analysis methods, which function both as prognostic and diagnostic tools.

Computational Hemodynamics (CHD) is the numerical simulation of blood flow in the different parts of the cardiovascular system. Blood Flow motion (e.g. the N-S equations) which is modelled by using CFD, and its interaction with the vascular structure, which is modelled by Finite Element Analysis are combined in such a way that they can provide a realistic simulation of the cardiovascular system function. These two fields (CFD and FEA) require real geometries to be solved, hence Medical Imaging for geometry reconstruction is of great importance. Figure 4.5 represents the correlation of the fields required for a complete CHD study.

Personalized numerical simulations of physiological processes in the human body have caught the attention of many during the last decades, since a great deal of models have been presented in literature [59]. We should not forget the impactful contribution of medical imaging to this new
field, as scientists now work on patient-specific models, and not on simple geometries like cylinders and ellipsoids.

4.2.1 Advantages of Computational Hemodynamics

Computational Hemodynamics is an emerging field that has already been applied in the virtual planning of surgeries, the evaluation of vascular diseases and in the development of medical devices. The drastic increase in computing power has made CHD a reliable tool for the experts that are now capable of simulating flow fields in complex geometries and regions where it’s impossible to conduct experimental measurements.

Regarding practical solutions that have been developed using CHD, the examples are countless. One can mention the coronary and endovascular stents for the treatment of coronary heart disease and the surgical treatment of aneurysms respectively, the artificial heart valves, the heart pumps, and finally even the total artificial hearts used for patients with biventricular failure. However, CHD is not only used in the industry but for research purposes as well. Several numerical studies have been carried out aiming to assess the mechanisms and flow conditions in a variety of diseases like the blood flow in aneurysms, in atherosclerotic arteries or even the heart chambers like this thesis deals with.

![Figure 4.8. Three applications of Computational Hemodynamics. (A) refers to coronary stent design [60], (B) is a ventricular assist device [61], and (C) is a simulation in a human aorta [62]](image)

4.2.2 Computational Modeling of Cardiac Hemodynamics

Cardiac Hemodynamics modelling is the cutting edge of numerical simulation since heart function depends not only on fluid dynamics but also on solid mechanics and electrophysiology as well. Therefore, in order to conduct a complete simulation, knowledge of the basic principles of all these scientific fields is required. Several heart diseases’ conditions can be simulated, such as cardiomyopathies, valvular diseases and arrhythmias (AFib).

However, although the possible applications might be countless, the simulation time can take up to days or even weeks, as the number of cells (millions) used and the complexity of the problems
is extraordinary high. Therefore, the need for parallelization and increase in the number of processors is of upmost importance if we want cardiac simulation to play a key role in the treatment of cardiac diseases and become a viable option for clinical doctors [68].

4.3 OpenFOAM

In this part of the thesis, a description of the main features of the software that will be mostly used for the solution of our problem is carried out. OpenFOAM ® [69, 70] (Field Operation And Manipulation) is an open-source fluid-simulation software that started its operation as FOAM back in 1993 (Imperial College – Henry Weller, Hrvoje Jasak). The goal was to create a robust CFD code without the disadvantages of the previous software developed in Fortran. FOAM remained inaccessible for public use until 2004, when the first open-source version was released. Since then, it has been used both in industry and the academic sector as well. Its basic advantage over other software, is its free use, making it possible for every user to modify the code according to his requirements, while at the same time members of the community can contribute to its development.

OpenFOAM is written in an object-oriented programming language, C++, providing greater flexibility to the user, creating executables known as applications. OpenFOAM, like Ansys Fluent as well, uses the finite volume method, while the flow magnitudes are defined at the centroids of the cells (Cell-Centered Method).

Moreover, OpenFOAM, which is contains around 100 C++ libraries, consists of pre- and post-processing environments. It comes with approximately 250 pre-built applications that are divided into two categories:

- Solvers, which are designed for the solution of a specific problem in fluid (or even solid) mechanics
- Utilities, that are used in order to perform tasks that involve data manipulation

Solvers included are suitable for a wide range of problems. However, users with some prerequisite knowledge in the specific physics of the problem and in programming, can modify the solvers or even the libraries. The overall structure of OpenFOAM is illustrated in Figure 4.8 [70].

![Figure 4.9. Overview of OpenFOAM structure [70]]
4.3.1 Solvers

The selection of the solver for each problem is an important step in the solution process, and it should be done considering the type and the underlying physics of the problem. The most widely used standard solvers for incompressible, compressible, and other types of flows are presented in Table 4.2.

<table>
<thead>
<tr>
<th>Solver</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>icoFoam</td>
<td>Transient, laminar flow for Newtonian fluids</td>
</tr>
<tr>
<td>pimpleFoam</td>
<td>Transient, turbulent flow of Newtonian fluids on a moving mesh</td>
</tr>
<tr>
<td>pisoFoam</td>
<td>Transient, turbulent flow, using the PISO algorithm</td>
</tr>
<tr>
<td>simpleFoam</td>
<td>Steady-state, turbulent flow</td>
</tr>
<tr>
<td>sonicFoam</td>
<td>Transient, trans-sonic/supersonic turbulent flow</td>
</tr>
<tr>
<td>rhoSimpleFoam</td>
<td>Steady-state, turbulent flow</td>
</tr>
<tr>
<td>interFoam</td>
<td>2 incompressible, isothermal immiscible fluids</td>
</tr>
<tr>
<td>mhdFoam</td>
<td>Laminar flow of conducting fluid under a magnetic field</td>
</tr>
</tbody>
</table>

Table 4.2. A sample of standard solvers used in different kinds of problems. Solvers in bold will be used in the present work

4.3.2 Standard “Utilities”

We will present the main functions of the utilities we used for the construction and solution of our problem.

- **blockMesh**: It is a multi-block mesh generator. We use it as an initial and auxiliary mesh that helps us generate the final mesh of our three-dimensional complex geometry. The initial mesh made, is actually a box that should contain the input model (e.g. 3D stl. model). More details about this will be presented in the following sections.
- **snappyHexMesh**: It generates the final mesh of the geometry using hexahedra (hex) and split-hexahedra (split-hex). It has a steep learning curve, as there are numerous options that should be set manually based on the user’s preferences. Will be also discussed in the mesh generation section of this thesis.
- **decomposePar**: This utility automatically decomposes a mesh and fields of a case for parallel execution. For large meshes with millions of cells, this utility is irreplaceable.
- **checkMesh**: It checks the validity of a mesh generated by either blockMesh or snappyHexMesh. Mesh generation criteria that have been set previously (like the aspect ratio or the skewness) are printed indicating whether they have been met or not.

There are other utilities as well, that will be used for our case, but they will be presented later in this thesis.

4.3.3 File structure of OpenFOAM cases

The typical file structure of OpenFOAM cases is presented. Each case consists of three basic files, the time, system and constant directories. Each one of them is of special importance for the case execution. The structure is as followed.

Department of Mechanical Engineering & Aeronautics - Division of Energy, Aeronautics & Environment
• 0/
• constant/
  o polyMesh
    ▪ faces
    ▪ points
    ▪ boundary
    ▪ ...
  o …Properties (…transportProperties, turbulenceProperties, …)
• system/
  o fvSolution
  o fvSchemes
  o Other dictionaries for utilities (decomposeParDict, createPatchDict, controlDict,…)

Directory 0/ includes individuals files of data for particular fields (e.g. velocity, pressure etc), that refer to initial values and boundary conditions that the user is required to specify for the definition of the problem. There are multiple boundary conditions to be selected (including Dirichlet, \( y(\alpha) = \alpha \), Neumann, \( y'(\alpha) = \alpha \) and mixed BCs). The user can also create his own boundary conditions as long as none of the pre-built satisfy his preferences (e.g. codedFixedValue – will be discussed later).

Directory constant/ contains dictionaries that define the fluid and flow properties (e.g. density, newtonian or no flow, laminar or turbulent, turbulent model, etc.) and also the subdirectory polymesh that includes the data structure of the mesh. The file “points” includes the nodes’ coordinates of the mesh, “cells” and “faces” files include information about the mesh cells (or elements) and faces, while in the folder “boundary” the type of every patch is defined (wall, patch, symmetry plane etc.)

Finally, in the directory system/ there are files (dictionaries) for setting parameters associated with the solution procedure. It should contain at least 3 dictionaries: controlDict, where run control parameters are set (e.g. start/end time, time step \( \Delta t \), etc.), fvSchemes, where discretization schemes are selected and fvSolution, where the equation solvers, tolerances and algorithm control are set.

Considering all the above, we can now proceed with an example of modifying the “source” code of OpenFOAM, based on our preferences in order to get a look of how solvers really work. An easy example to understand the basic principles, is to add temperature to the simplest solver, icoFoam. We will actually add the heat transport equation (Eq.(4.14)), that can be solved simultaneously with the other fields (velocity, pressure, etc).

\[
\frac{\partial T}{\partial t} + (\nabla \cdot V)T = a\nabla^2 T, \quad (4.14)
\]

where \( a \) is the thermal diffusivity index.
Before changing anything, we can get a glance of how the source code of icoFoam looks. We simply search for the icoFoam.C file that is saved in the OpenFOAM installation directory. In this directory none of the files can be modified, yet they can be copied to our home directory and get renamed so we don’t confuse them with the unmodified ones. Figure 4.10 represents a thorough view of the solver itself.

It can be seen that for the solution process of two fields only, we have two equations, Ueqn and Peqn. IcoFoam includes PISO algorithm which will be discussed in the next subchapter. We can clearly notice the Ueqn in which fvm means that the finite volume method is used (fv) and m means that the discretization is implicit. Fvc considers explicit discretization for pressure gradient. Finally, phi refers to the flux as we know it in the finite volume method, and nu is the kinematic viscosity $\nu$. What is apparently being solved is the N-S equations in the form below.

$$\frac{\partial}{\partial t} (U) + \nabla \cdot (UU) - \nabla \cdot (\nu \nabla U) = -\nabla p,$$  \hspace{1cm} (4.15)

where $U$ is the velocity and $p$ the pressure.

In the beginning of the file before the solution loops, some headers are included, with the most important for us being the createFields header. This is a file which contains information about the variables that will be solved. It is shown in Figure 4.11.

Details about the scalar pressure field, the vector field of velocity and the kinematic viscosity are included. Therefore, we have to add the thermal diffusivity $\alpha$ (Eq.4.14) in the same exact way as nu is added, below nu. Afterwards, the transportProperties dictionary of our case must be edited.
In addition, in the same way pressure and velocity fields are included, we add a second scalar field (volScalarField) that represents the temperature.

Finally, what’s left to be done, is to add our new equation describing the transport of the temperature. As temperature transport relies on the velocity field, the equation is added after the PISO loop, and before the time step is written. We have now successfully added the temperature field. Both createFields.H and icoFoam.C final versions will be included in the Appendix.

Since we are not going to solve a case depending on temperature, neither boundary conditions will be presented nor the numerical schemes and solvers.

4.3.4 The numerical algorithms: SIMPLE, PISO and PIMPLE

In order to solve the Navier-Stokes equations, some numerical techniques are required capable of solving this coupled pressure-momentum system. These algorithms are known as SIMPLE, PISO and PIMPLE.

A summary of under which conditions these algorithms are used is shown below:

- **SIMPLE** (Semi-Implicit-Method-Of-Pressure-Linked-Equations). We use this algorithm in OpenFOAM for steady-state simulations
- **PISO** (Pressure-Implicit-of-Split-Operations). It is used for transient simulations. It is strictly based on the time step selected, which is associated with the Courant Number (Co<1)
- **PIMPLE** (Merged PISO-SIMPLE). It combines the mentioned algorithms, providing the user with the option to use larger time steps (Co ≫1)
Solvers based on \textit{SIMPLE} algorithm, don’t include time derivation in the equations, therefore we can only calculate a steady-state solution. SIMPLE algorithm is not consistent in terms of numerical analysis. In OpenFOAM there is a modification of the algorithm, which makes it consistent, and it is known as SIMPLEx. The fact that the lack of the natural limiter to a solution, \( \Delta t \), and at the same time the inconsistency of the method, leads to the need for under-relaxation of the fields (e.g., U, p) in order to reach convergence and achieve stability. If no under-relaxation is performed, the solution might blow up leading to an unphysical solution \cite{71}.

In contrast, the \textit{PISO} algorithm, has two main differences compared to SIMPLE. It includes the time derivation term and the pressure-velocity coupling equation is consistent. Therefore, there is no need for under-relaxation, but a stability criterion must be met. The latter indicates that the Courant number (Eq.(4.16)) must be lower that one.

\[
Co = \frac{U \Delta t}{\Delta x} \leq 1, \tag{4.16}
\]

where \( U \) is the velocity, \( \Delta t \) is the time step and \( \Delta x \) the distance between cells centroids. In OpenFOAM \( \Delta x \) actually refers to the volume of the cell.

If the Courant number is under one, the information from one cell can only reach the neighbor to itself within one time-step, whereas if Co is greater than one, the information can reach more than one cells, which is not allowed based on some explicit aspects.

Relying on the definition of the Courant number, assuming that we refine the mesh (more cells-smaller volume) and increase the velocity or the time-step, the value will increase. Consequently, we must adjust the time-step in each case depending on the mesh size and velocities. The Courant number criterion is performed for every single cell of the mesh, and therefore if one cell fails to achieve this number, it is very likely that it will affect the whole simulation.

Finally, the \textit{PIMPLE} algorithm is one of the most commonly used algorithms for transient simulations, since it combines features from both PISO and SIMPLE algorithms. The time step this method implies can be increased leading to lower simulation time required, as it’s possible to receive greater Courant values (\( Co \gg 1 \)). The mechanism of the algorithm starts with a time step during which a steady-state solution is searched while at the same time an under-relaxations strategy is used. Before going to the next step, we acquire the outer correction loops ensuring that all explicit parts of the equations have converged. After the tolerance criterion defined by user has been reached in the steady-state calculation, the outer correction is over and moving onto the next time step is possible. This is continuously done until the simulation time ends.

Bearing in mind all the above, we can now proceed with the pre-processing of our simulation of blood flow in the left atrium. All the necessary steps will be presented including the processing of the 3D model, including 3D Slicer, Blender and VMTK. Ansys Fluent will be used as well, in order to compare the generated Mesh to OpenFOAM’s one, however the simulation will be exclusively conducted in OpenFOAM.
5 Pre-Processing

In this chapter of the present thesis, we are going to present the procedure which is required for preparing the geometry that will be used for the CFD simulation. Pre-processing is a very important step, since the quality of the geometry, which depends on the initial imaging process, the construction of the 3D model and the mesh generation, affects the final results of the simulation.

5.1 Reconstruction of the 3D model

The first step we are going to focus on is the 3D model reconstruction from the MRI data. What we are offered for the conduction of this thesis is an unprocessed model of the Left Atrium that is generated directly from MRI images as shown in Figure 5.1. The method used by the doctors is called Phase-Contrast MRI, which apart from the obvious application in imaging, it can also visualize and quantify the velocity field. These data can be used for defining the boundary conditions of the inlets and outlets, which will be discussed later. In Figure 5.1 the 3D model is accessed through 3D Slicer ® (a free software for visualizing and analyzing medical image computing data sets [72])

![3D model reconstruction from Phase-Contrast Magnetic Resonance Imaging. MRI sample image obtained from [73]](image)

After opening our 3D model in 3D Slicer as a segmentation, we can see the interface of the software. We can notice that the model is divided into the blood and the wall domain, as during the image processing it was possible to detect and separate the wall out of the fluid region. In Figure 5.1, yellow indicates the blood volume and gray represents the LA wall. In addition, we can clearly observe the 4 pulmonary veins and the LA appendage. The mitral valve area is within the range of 4 to 6 cm². Afterwards, we are going to keep the blood volume segmentation as we are going to simulate and generate the mesh for the fluid region only. In case we wanted to carry out a Fluid Structure Interaction (FSI) simulation, details about the wall would be required as well.

Nevertheless, this is not going to be the final model, since it should be smoother and sharper. In the meantime, it should be cropped in such a way that we have our patches open and “circular”.

Department of Mechanical Engineering & Aeronautics - Division of Energy, Aeronautics & Environment
This is done in 3D Slicer as well. We then use the segment editor tool and then scissors effect to cut anything needed and remove it by clicking on keep largest island.

After processing the geometry so that the inlets have a circular shape, we save it as an STL file. We proceed with the smoothing of the geometry in order to look less layer-like and have sharp details. We will use Blender®, a free and open-source software for 3D modeling [74].

Importing the STL file into Blender, we select the object mode and then sculpting. We will use the smooth and flatten features to process our geometry. Moreover, as the region of the mitral valve, which is included in the blood volume domain, is not well shaped we will use the filling option. In addition, considering the existence of some irregular “peaks” in the 3D model, we apply the dyntopo filter to avoid them, since such irregularities can lead to flow problems (e.g. blood stasis and dramatic increase in the ECAP and RRT indices).
Following the smoothing of our geometry, we advance with the final software called VMTK® (the Vascular Modeling Toolkit [75]) which is a collection of libraries and tools for 3D reconstruction, geometric analysis, mesh generation and surface data analysis for image-based modeling of cardiovascular geometries. We will use VMTK for two reasons. First is to cut the inlets and outlets so that they suit our preferences for mesh generation and flow conditions, and the second is to extend the inlets either to achieve a fully developed flow or shape them exactly circular so we can acquire a parabolic velocity profile.

The VMTK toolkit is based on Python programming language, however no specific knowledge is required. For every feature available, there is a special command provided on the toolkit’s website. First of all, since our surface is closed at inlets and outlet with a blobby appearance, we will crop them as vertically as possible, so that we receive a circular patch, except for the mitral valve (the outlet) which has an oval shape. Figure 5.4 illustrates the user interface of the toolkit, and also the command needed for the initialization of the clipping procedure (geometry_from_blender.stl is the stl file extracted from blender and clippedLA.stl is the model after the procedure).

Figure 5.4. User Interface of VMTK toolkit

Figure 5.5. The clipping procedure
After clipping every patch, we can continue with the flow extension of the inlets. We will present both options, one with fully extended inlets and the other with an extension small enough to form circular inlets.

To begin with, flow extensions are just cylindrical extensions to the inlets of the model. This is a significant process as it ensures that the flow entering the domain is fully developed. As presented in the previous chapter, Eq. (4.4) implies that for laminar flow, as we are certain that the pulmonary veins are so small compared to the other parts of the domain and the velocity is not very high, that the length required for a fully developed flow is \(0.05 \cdot Re \cdot D\). Therefore, assuming that the Reynolds number is in the worst case 2200, the maximal diameter of our PVs is 0.015m, the ratio \(L/D\) would be approximately 100. However, this is impossible to achieve, as it would require a lot of area. At the same time if the flow was assumed to be turbulent, then the ratio \(L/D\) would be around 10 for a fully developed flow. Hence, we will only acquire a ratio \(L/D\) close to 20 and check how it looks. The simple command is given in the Appendix II together with any other used.

We can clearly see that the addition of flow extensions with a ratio of \(L/D = 20\), would not be the ideal solution for a acquiring a fully developed flow (with a uniform velocity profile for the inlets), as the number of cells that would be required for the mesh generation, would possibly double and drastically increase the simulation time. The second idea is to introduce a parabolic profile normal to the circular inlets, instead of extending the latter and increasing the mesh size. Therefore, we would only need a very small extension, such as \(L/D =1.5\), which is enough to turn the different profiles of the inlets to exactly circular. However, this idea is not the best as well, as we should code ourselves the boundary condition of the parabolic profile at the inlets, while bearing in mind, that even the parabolic profile, is not always what occurs in reality, since the curvature of the pulmonary veins (and the branches before they turn to RSPV, LSPV etc) returning the blood from

*Figure 5.6. Added flow extensions \((L/D =20)\) to the inlets*
the lungs, may lead to different velocity profiles, with flows more attached to the wall for instance. Still, this is the closest to reality option, while maintaining a relatively normal simulation time. Eventually, we will use VMTK again and extend the inlets only with a factor of 1.5. Afterwards, we will use Blender to smooth any irregularities until our model takes its final form, which is presented in Figure 5.7.

![Figure 5.7. Final form of the LA geometry](image)

After converting the dimensions of the model into the correct ones of the patient (mm to m) we can proceed with the meshing procedure using the snappyHexMesh utility of OpenFOAM. We should also mention that snappyHexMesh requires a capped version of the surface meaning closed inlets and outlets, so we close them with the VMTK feature.

### 5.2 Meshing Procedure

In this part of the thesis, the mesh generation procedure will be presented, including a description of every utility used in the process.

#### 5.2.1 BlockMesh and surfaceFeatureExtract Utilities

First of all, the OpenFOAM version that is used for this work is the 7th, so possible changes coming with the newer versions (9th) might affect the future readers of this thesis. The first step required is to create a grid that contains our 3D model, which is done by editing the blockMesh Dictionary located in the /system directory. We actually create a box (block) that is meshed with a specified number of cells, neither too many nor too few. On each side of the three-dimensional block, we select the number of cells. Each block has a local coordinate system (x1, x2, x3) that is right-handed. Finally, a rectangular grid 35x35x35 is created after setting such coordinates that include our geometry \((x_0 \rightarrow x, y_0 \rightarrow y, z_0 \rightarrow z) = (-0.06 \rightarrow 0.06, -0.01 \rightarrow 0.062, -0.06 \rightarrow 0.03 \text{ m})\). Since our geometry doesn’t have any long cylindrical extensions that could be intersected by this block, in order to be defined as patches, none of the block’s patches are of our interest (if one side of the box was intercepted by an extension of RSPV for instance, we would later define it as an inlet patch). Figure 4.8 depicts the blockMesh dictionary and Figure 5.9 the grid including the LA model.
Figure 5.8. BlockMesh Dictionary

Figure 5.9. Generation of the block containing the geometry
With the execution of the blockMesh utility the polyMesh file is created in the /constant directory including data about the boundaries, the faces of the mesh created. In the same file there are already located the triSurface file, where our 3D model is saved into and as previously mentioned, the files about transport and turbulence properties, which will be discussed later.

Following the creation of the initial grid, we proceed with the execution of the surfaceFeatureExtract utility which extracts and writes surface features to a file. The dictionary for this utility is located in the /system directory. Within this dictionary (which is not provided as few changes are only made), we first specify the name of our STL model and the extraction method, which for our case happens to be a surface (we use the extractFromSurface option). Afterwards, we can set up the following entries that affect the behavior of the extraction. We select the includedAngle value as 120 degrees, that means that edges whose adjacent surface normal are at angle less that the specified value, are marked. Finally, the initial set of edges can be subsetted: we don’t keep nonManifold edges (the edges with more that 2 connected faces where the faces form more than two different normal planes) while keeping open edges which are edges with 1 connected face.

5.2.2 SnappyHexMesh Utility and parallel run

One of the most critical, perhaps the most critical, dictionaries is that of the snappyHexMesh utility. The meshing process is of paramount importance for the solution of every problem, and it is the basis upon which we obtain appropriate results, in addition to the solver itself. A good mesh in terms of quality mostly, is responsible for producing accurate results, and it’s the most probable cause that we should investigate in case our results are illogical. As mentioned, the snappyHexMesh feature involves a steep learning curve, and a lot of investigation of the options by the user itself. However, after dealing with its structure and use, each user is capable of understanding a variety of topological and geometrical issues coming with the generation of a 3D mesh. Since there isn’t a user-interface option (a “disadvantage” of OpenFOAM) the user must spend a lot of time coping with the trial-and-error method. Nevertheless, a complete control of the mesh features is available to the users.

To begin with, the SnappHexMesh dictionary is divided into three main features (steps):

1) the castellatedMeshControls
2) the snapControls
3) the addLayersControls

There are also some minor features (minor as for their complexity), which is the “geometry” and the “meshQualityControls”.

5.2.2.1 Geometry and CastellatedMeshControls

First comes the geometry subdictionary (or section) in which the geometry is specified through a tri-surface or bounding geometry entity. Therefore, for our case we specify the name of the STL file which is saved in the triSurface folder in /constant directory and a bounding box using the minimal and maximal coordinates of it (similar to the blockMesh coordinates). After specifying the geometry features, we advance with the castellatedMeshControls and the splitting of the cells at the feature edges and surfaces.
**Local and Global Cells**

First thing to be done is to define the number of cells that each processor of our system should deal with (maxLocalCells). As it will be mentioned again later, we are going to use 45 threads (or processors) for our simulation (including the meshing process), so there is no need to overload the processors with a lot of cells. For our case we set this number as 1000000, which is still high and there is no way that it will be exceeded. Next, we define the overall cell limit (maxGlobalCells) during refinement and that is before the removal occurs. We set this number to 2.5 million cells, since larger values of mesh sizes will increase the simulation time exponentially as long as our operating system does not consist of hundreds of processors.

**Features**

The splitting of the cells is considered to begin with the “features” option in the castellatedMeshControl section in which cells are selected according to specified edge features within the initial domain. In this option we define the name of the edgeMesh (.eMesh) file which is automatically generated with the execution of the surfaceFeature utility. In the same area, the level of refinement is specified. We use this option: (1 4), which indicates the use of 1 to 4 refinement levels (1 is the bigger cell size and 4 the smaller used near to the walls).

**Refinement Surfaces, Resolve Feature Angle and Refinement Regions**

Then, we set the refinementSurfaces option, in which we specify the min and max refinement level. The minimum level is generally used across the whole surface of the model, whereas the maximum is used in specific areas indicated by the resolveFeatureAngle. ResolveFeatureAngle controls the geometry curvature refinement. If the angle of the geometry at a specific area is higher than resolveFeatureAngle, the adjacent STL faces will be refined, whereas if it’s lower, no refinement takes place. Generally, the lower the value the higher the number of cells generated. A typical value is 30°. Finally, the refinement Regions option, requires the type of refinement and the refinement levels. Inside mode is used for the type and the levels are specified as (1e15 4) in which the first number is the distance from the surface, which has to be a large value to cover the entire region, and the second value is the refinement level.

![Figure 5.10. Mesh generated with castellatedMeshControls only turned on](image-url)
We should not forget to mention that as a final step the location of an internal point of our geometry is required, so it can be defined that the removal of cells will be affecting the outer cells, and not those inside the region of our model. In Figure 5.10 a mesh produced only with the castellatedMeshControl turned on is presented. The cells surrounding the model and generated the first time by the blockMesh utility have been removed, however it is quite clear that the edges are not smooth at all. In order to fix this problem, we have to turn on and edit the snapControls subdictionary.

5.2.2.2 SnapControls
Aiming to crop or snap the cells that lead to this “rough” surface, we edit the snapControls section. In general, only some settings can be modified, and they mostly depend on the trial and error method and our preferences. The first feature is called nSmoothPatch and it defines how many times the patches will be smoothed before correspondence to surface is found. In our case we set this number of iterations to 2. The second most important feature is nSolveIter, which specifies the number of mesh displacement relaxation iterations as the vertices in the castellated boundary are displaced onto the STL surface. The process includes solution for the relaxation of the internal mesh with the latest displaced boundary vertices. In order to improve the quality of the edge features we set this value to 50. The rest of the options which are less important will be presented in Table 5.1, which includes the majority of the values inserted. One more important step in the snapcontrols section is to define the type of featureSnap. After some tries, we concluded that the best option for our case is the implicitFeatureSnap which detects (geometric only) features by sampling the surface. Figure 5.11 illustrates the mesh generated with both castellatedMeshControls and snapControls turned on. We can now clearly observe the smoothness of the generated mesh compared to the previous one. The current mesh consists of approximately 1.9 million cells, which indicates the importance of parallel running. Details about the parallelization of the simulation will be presented after completing snappyHexMesh features.

Figure 5.11. Mesh generated with both castellatedMeshControls and SnapControls turned on
5.2.2.3 Add layers and Mesh Quality Controls

The addition of boundary layers to our mesh is of critical importance, since the main purpose of this thesis is to capture efficiently the wall shear stress distribution. The more boundary layers added the higher the accuracy of WSS values. However, this addition comes with a cost, meaning that more layers add more cells, hundreds of thousands for every layer in our case. The number of layers will be defined by a mesh independence study, which will be carried out in the next chapter.

The process under the layer addition is conducted includes shrinking of the existing mesh from the boundary and the insert of layer cells. In order for this feature to work, some criteria should be satisfied. First, we must specify 2 out of 4 different layer thickness parameters (expansionRatio, finalLayerThickness, firstLayerThickness, thickness). We set the expansion ratio value to 1, indicating that all the layers will have the same thickness. For instance, if we set that value to 2, every layer would have twice the thickness compared to the layer below. We also specify the final layer thickness to 0.25, which is relative to the undistorted size the cell outside of the layers, (after setting the option relativeSizes to true) indicating the thickness of the last layer (first is that near the wall). The mentioned options and the other two that were not selected can be understood by taking a look into Figure 5.12.

Moreover, the number of boundary layers is selected by the user. In the mesh independence study, we are going to use from 0 to at least 4 layers to check the accuracy of the solution. There are also several other parameters to be set, yet, without affecting severely the quality. One option that has taken a lot of time to be set correctly has been the featureAngle which specifies the angle above which the surface is not extruded. Although a typical value for this feature is around 120o, after a great deal of trial and error executions, we concluded that the best value is 30. A different value reduced the quality of our inlets and outlet as shown in Figure 5.12. This bad quality at the inlets and outlet lead to destruction of the solution, providing wrong results. As mentioned before, the mesh quality is largely responsible for the final results. In order to solve this issue, a lot of time had to be spent trying to find the cause, which finally an addLayers option. The addition of layers, except for increasing the mesh size, can also cause a variety of problems related to the quality of mesh.

Figure 5.12. Boundary layer parameters set on addLayers section [76]
**Figure 5.13.** Irregularities fixed by changing featureAngle value

**Figure 5.14.** Irregularities at the patches section caused by the featureAngle value
Before generating the mesh, there is a number of quality parameters that are required to be set, satisfying our preferences. First comes, perhaps, one of the most important criteria for mesh generation that is known as maxNonOrtho, and specifies the maximum non-orthogonality allowed. Typically, the value used is 65 (we also used this one), and values larger than 80 make it hard to achieve a convergent solution [77], so basically the value of this term affects our solution. As we will see later in the solution method, in our numerical schemes we include non-orthogonal correctors. A similar to non-orthogonality parameter is skewness which was also discussed previously. We set the internal skewness value to 4 and the boundary skeweness value to 20. Figure 5.14 presents the difference between a skew and a non-orthogonal mesh.

![Figure 5.15. Difference between non-orthogonal and skew mesh [78]](image)

There are also several other parameters, but their values will be only presented in Table 5.1.

### 5.2.2.4 Parallel Run

It is significant to bear in mind that in cases where high detail in both the geometry and the results is required, the solution time can be extremely large. The required accuracy leads to a very high number of cells and especially in geometries such as those of biomedical applications, the number of cells of the grid can reach up to some millions, depending on the preferences of each user. This way, we realize the necessity of performing the mesh generation and the solution process in more than one thread of our system’s processor.

In this parallelization process, the domain containing either our geometry only or flow fields as well, is decomposed or broken into pieces and allocated to separate files, one for each processor (or thread). The parallel running process is conducted by using the public domain openMPI implementation of the standard message passing interface (MPI).

The decomposition is done using the decomposePar utility. There are four different ways of decomposition: simple, hierarchical, scotch and manual. As shown in Figure 5.16, simple method is used for our case, which defines that the domain is split into pieces by direction. In our case 45 processors are used, so as described in Figure 5.16 mesh is split into 5 parts in x direction, in 3 parts in y direction and 3 parts in z direction.
An example of how a simple block is decomposed is displayed in Figure 5.17. The domain is divided in 45 parts, 5 in x direction, 3 in y, and 3 in z.

After the execution of the command “decomposePar” followed by the parallel run for the mesh generation (“mpirun -np 44 snappyHexMesh -overwrite -parallel”) we can see how quickly most of the processors are getting used.
We have now finished the meshing part. In Table 5.1 the most significant parameters inserted in SnappyHexMesh dictionary are given.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>castellatedMeshControls</td>
<td></td>
</tr>
<tr>
<td>maxLocalCells</td>
<td>1000000</td>
</tr>
<tr>
<td>maxGlobalCells</td>
<td>2500000</td>
</tr>
<tr>
<td>nCellsBetweenLevels</td>
<td>1</td>
</tr>
<tr>
<td>features levels</td>
<td>(1 4)</td>
</tr>
<tr>
<td>resolveFeatureAngle</td>
<td>30</td>
</tr>
<tr>
<td>refinementRegions</td>
<td>refinementBox levels (1E15 4)</td>
</tr>
<tr>
<td>snapControls</td>
<td></td>
</tr>
<tr>
<td>nSmoothPatch</td>
<td>2</td>
</tr>
<tr>
<td>Tolerance</td>
<td>10</td>
</tr>
<tr>
<td>nSolveIter</td>
<td>50</td>
</tr>
<tr>
<td>nRelaxIter</td>
<td>10</td>
</tr>
<tr>
<td>nFeatureSnapIter</td>
<td>10</td>
</tr>
<tr>
<td>implicitFeatureSnap</td>
<td>true</td>
</tr>
<tr>
<td>addLayers</td>
<td></td>
</tr>
<tr>
<td>nSurfaceLayers</td>
<td>4</td>
</tr>
<tr>
<td>expansionRatio</td>
<td>1</td>
</tr>
<tr>
<td>finalLayerThickness</td>
<td>0.25</td>
</tr>
<tr>
<td>featureAngle</td>
<td>30</td>
</tr>
<tr>
<td>nLayerIter</td>
<td>50</td>
</tr>
<tr>
<td>meshQualityControls</td>
<td></td>
</tr>
<tr>
<td>maxNonOrtho</td>
<td>65</td>
</tr>
<tr>
<td>maxInternal/BoundarySkewness</td>
<td>4/20</td>
</tr>
<tr>
<td>minFaceWeight</td>
<td>0.05</td>
</tr>
<tr>
<td>minVol</td>
<td>1e-16</td>
</tr>
</tbody>
</table>

*Table 5.1. Main parameters for a good quality mesh*
Figure 5.19 represents the meshing procedure followed for our model.

5.2.3 Definition of Patches – topoSet and createPatch Utilities

Although we have created the mesh, as someone may have noticed, no inlet and outlet patches are defined at the moment. The whole surface of the model corresponds to the wall patch. Irregular geometries such as our own, are not easily meshed using snappyHexMesh utility since there is no graphic user interface provided. Therefore, different ways are proposed for defining the patches as inlets and outlets. One is to add flow extensions, as discussed previously, for acquiring fully developed flow. In this way, one side of the block (see Figure 5.9) would cut the extended cylinder and the cross-section created would be then defined as a patch, which would be later specified as inlet. However, this is a solution mostly preferred for blood vessels, which in reality are quite long.
themselves (e.g. aneurysms, and even when extended) and all the patches can cut one side of the initial block. In case that we would not like to extend each patch, for example the outlet, we would use topoSet utility (will be discussed shortly). In our case, the idea of extending the inlets in order to cut the initial blockMesh is not acceptable since it would drastically increase the mesh size and at the same time, the patches would be rather elliptical and not circular as we actually want, preventing us from adding the parabolic velocity profile.

Aiming to solve this issue, since nothing similar could be found on bibliography, we introduced a new method using the utilities topoSet and createPatch.

TopoSet utility collects a region of the created mesh cells or even faces that can be then handled in different ways. What we aim to do is to generate small volumes (could be boxes, cylinders, etc) that contain the inlet part of the patch and by keeping those faces which are normal to the volumes created, we can later define them as patches. Figure 5.20 shows the topoSet dictionary including the three steps required for maintaining only the set of faces which define the circular region.

[Image of topoSet dictionary sample]

The first step is to create a cylinder (source: cylinderToFace) containing the region where our desired inlet is. For this reason, we specify the cylinder’s radius and the coordinates of the two endpoints. This is a time-consuming procedure since the coordinates have to be specified manually as there is no user interface and obviously there is no connection between the dictionaries and the paraView environment.

Afterwards, for this set, we create a subset that selects faces on all external boundaries. By applying these two actions we receive the produced inlet region after running createPatch utility which simply creates patches out of selected boundary faces (those created by topoSet). Both dictionaries, topoSet and createPatch are located in the /system directory. After running createMesh utility, a file named with a number (typically the time step value) is created in the main directory of our case, containing the new polyMesh file which must replace the existing one located in /constant directory. Figure 5.21 shows how the inlet RIPV looks like after applying the cylinderToFace and boundaryToFace features.
The result obviously does not satisfy our criteria as the inlet must be defined normal to that surface and should not include any of the wall region. To solve this issue, we used another subset in addition to the previous, named normalToFace, that selects faces whose surface normal is aligned with a given direction. That means that we have to find the normal vector of each patch, for both inlets and outlet. Again, this procedure is not easily conducted, so we had to find a way manually using openFOAM’s internal data for the mesh. As will also be presented later for acquiring the parabolic profile to the inlets, we use codedFixedValue type of boundary conditions for velocity field, located as U in the 0/directory. In OpenFOAM, for every single face, there is information about its numbering in the entire mesh (e.g. face #120 out of 150) and most importantly information about its normal vector. By acquiring a loop for every patch asking for information about the normal vector (info<<nHat<<endl) a list of the vectors for every face is printed in the command prompt. Remember that there are also those faces in the wall region, however we can easily understand that those in the desired patch would have almost the same normal vector (given as \((x,y,z)\)).
We can see that the faces’ normal vectors are almost identical, so they are certainly those covering the desired patch area. Finally, in the normalToFace subset, we insert the normal vector dimensions (see Figure 5.20) and we also specify a tolerance angle, which is the cosine of angle between the input normal and the face normal, and it determines whether some faces with similar dimensions should be kept or not.

The described procedure is done for every patch of our model, so we receive similar results to the one shown in Figure 5.23.

![Figure 5.23. Final definition of patches](image)

5.2.4 CheckMesh Utility

Finally, before advancing with the solution, we should check whether the generated mesh satisfies the criteria defined in the meshQualityControls section in snappyHexMesh dictionary and in the meantime, receive a detailed view of the mesh characteristics such as the number and the type of cells used.
After executing the checkMesh utility we receive the following data for a 4-layer mesh.

<table>
<thead>
<tr>
<th>Mesh Generated using SnappyHexMesh</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Number of Cells</strong></td>
</tr>
<tr>
<td><strong>Hexahedra</strong></td>
</tr>
<tr>
<td><strong>Prisms</strong></td>
</tr>
<tr>
<td><strong>Tet Wedges</strong></td>
</tr>
<tr>
<td><strong>Tetrahedra</strong></td>
</tr>
<tr>
<td><strong>Polyhedra</strong></td>
</tr>
<tr>
<td><strong>Patches</strong></td>
</tr>
<tr>
<td><strong>LA #faces</strong></td>
</tr>
<tr>
<td><strong>RIPV #faces</strong></td>
</tr>
<tr>
<td><strong>RSPV #faces</strong></td>
</tr>
<tr>
<td><strong>LSPV #faces</strong></td>
</tr>
<tr>
<td><strong>LIPV #faces</strong></td>
</tr>
<tr>
<td><strong>Mitral #faces</strong></td>
</tr>
<tr>
<td><strong>Mesh non-orthogonality Max/Average</strong></td>
</tr>
<tr>
<td><strong>Max skewness</strong></td>
</tr>
<tr>
<td><strong>Max aspect ratio</strong></td>
</tr>
</tbody>
</table>

Table 5.2. Mesh details after running checkMesh utility

All the above characteristics lead to an “OK” message after executing checkMesh utility, meaning that the mesh satisfies the defined criteria. The number of cells is extremely high leading to large computation time not only for the generation of mesh (approximately 240 sec, using 45 processors), but mostly for the solution process, that may even take up to 10 days based on empirical data. In addition to the checkMesh message, we can also gain data about the layer addition process and how successful it was, in the end of snappyHexMesh run. For the addition of 4 layers, we managed to get a 94.9% coverage with an average addition of 3.82 layers, which are certainly efficient for capturing wall shear stress values.

Nevertheless, a mesh independence study will be carried out later in the solution chapter, where the appropriate mesh size will be chosen. As mentioned previously, the mesh without the boundary layers consists of approximately 1.9 million cells, which is still a large value, so in order to carry out a complete mesh independence study, which will not only contain a comparison between the addition of boundary layers but also different refinement levels and cells volumes (~300k, ~2m, ~8m cells), we are going to generate a mesh with a size under one million using the Ansys Fluent Mesher.

**5.2.5 Mesh generation in Ansys Fluent (Academic Version)**

Furthermore, as we also want to present the convenience under which a mesh can be generated in a commercial software with a powerful graphical user interface, and at the same time produce a mesh with less than 500000 cells (this is the limit for the academic version) for our independence study, we will use the Ansys Fluent ® meshing tool. The generated mesh will be then exported in a proper format, so it can be used for the solution in OpenFOAM.
In the first place, it could be said that the meshing procedures are quite similar, however, an initial grid like this created by blockMesh utility is not needed in Fluent. This allows for a much easier creation of sets (similar to topoSet) that helps us define our patches. It should be mentioned that for the mesh generation process in Ansys Fluent, the STL model must be open, and not capped in contrast to snappyHexMesh.

The meshing tool that we are going to use is called Shrink Wrap, as it provides a convenient solution to problems like this. After importing the geometry into fluent, we create a bounding box that contains our geometry. Since our geometry is not “water tight” in order to shrink wrap the internal volume, we create cylinders using the “create object” feature that cover the desired patches as shown in Figure 5.25. We then rename the surfaces, of the volumes made, that cut our patches and finally we set the boundary type. The second step is to create a material point within the geometry and name it as blood. This will be used later for the shrink-wrap step. The following steps refer to mesh characteristics and will not be presented in this work. A complete guide is presented in [79].

![Figure 5.25. Use of cylinders and bounding box for the shrinkwrap process in Ansys Fluent meshing tool](image)

It should be noted that in Fluent, we are capable of using tetrahedra, hexahedra and most importantly polyhedral cells. Polyhedral meshes, can be certainly considered as a viable alternative to tetrahedral meshes, as their computational convergence has improved. This leads to a smaller mesh size, compared to the tetrahedral one, reducing the computational cost. As described by Marti Spiegel et al [80], polyhedra share information with more neighbors than tetrahedral or hexahedral (produced by snappyHexMesh) and in accordance with all the above, polyhedral meshes should be preferred over tetrahedral meshes for clinical CFD simulations. The utilization of the generated polyhedral mesh will for sure lead to an interesting mesh independence study in the next chapter.

Figure 5.26 shows the generated mesh of our LA geometry. It utterly clear how smooth and uniform the cell topology has been conducted, as we make out that the use of different polyhedral cell sizes in different areas of the model where more detail is required.

As for the main characteristics of the mesh, it consists of 328885 cells (with a maximum of 500k that can be generated) and 5 boundary layers. It contains only 23942 hexahedra and 304943 polyhedra (in contrast to the mesh produced by snappyHexMesh, see Table 5.2). All the details

Department of Mechanical Engineering & Aeronautics - Division of Energy, Aeronautics & Environment
are included in Table 5.3. Finally, what’s interesting and in the meantime justifies the superiority of polyhedral mesh, is the faces per cell. For this mesh, the cells include on average 9.3 faces (and therefore share neighbors) in contrast to the 6.1 faces in the mesh generated by snappyHexMesh.

<table>
<thead>
<tr>
<th>Mesh Generated using Ansys Fluent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Cells</td>
</tr>
<tr>
<td>Hexahedra</td>
</tr>
<tr>
<td>Prisms</td>
</tr>
<tr>
<td>Tet Wedges</td>
</tr>
<tr>
<td>Tetrahedra</td>
</tr>
<tr>
<td>Polyhedra</td>
</tr>
<tr>
<td>Patches</td>
</tr>
<tr>
<td>LA #faces</td>
</tr>
<tr>
<td>RIPV #faces</td>
</tr>
<tr>
<td>RSPV #faces</td>
</tr>
<tr>
<td>LSPV #faces</td>
</tr>
<tr>
<td>LIPV #faces</td>
</tr>
<tr>
<td>Mitral #faces</td>
</tr>
<tr>
<td>Mesh non-orthogonality Max/Average</td>
</tr>
<tr>
<td>Max skewness</td>
</tr>
<tr>
<td>Max aspect ratio</td>
</tr>
</tbody>
</table>

*Table 5.3. Mesh details after running checkMesh utility for Ansys Fluent generated mesh*

*Figure 5.26. Polyhedral Mesh generated using Ansys Fluent Meshing tool*
6 Solution Method

In this part of this thesis, the steps required for the solution process will be presented. This involves the solvers and the numerical schemes acquired and also the boundary conditions that will be implemented for the different cases that we are going to run.

6.1 Solvers and Numerical Schemes

First and foremost, we should decide the type of flow that we aim to use. The main simulation for which we expect the results regarding velocity profiles and wall shear stress distribution, includes a transient flow. For this type of flow, we are going to utilize the pisoFoam solver (see Table 4.2), which has been proven to be stable enough for transient problems involving a newtonian fluid. In addition, its main advantage is that it can simulate both laminar and turbulent conditions, in contrast to the icoFoam, which seems to be a simpler solver. Although there is proof [ ] (see Table 6.4) that the flow in the pulmonary veins and probably within the left atrium as well, is mostly laminar we are going to compare laminar and turbulent conditions using a variety of turbulence models such as k-Epsilon, k-Omega-SST, and LES, whose parameters will be analyzed. However, steady-flow conditions will also be implemented in order to conduct the mesh independence study for which different mesh sizes will be used. The implementation of steady flow leads to a less time-consuming process compared to using transient flow, allowing us to carry out this process using a lot of different meshes and comparing different turbulence models.

We can now proceed with our steady-state solver, simpleFoam, examining the files related to it, located in the /system directory. First of all, in order to determine the solver that will be used for the simulation, we specify its name in the control dictionary in the application section. We also specify the time step, which is of paramount importance especially for the transient-flow solvers. For steady flows we usually define the time step as 1, and then we specify the number of iterations in order to check whether the model has converged to a solution.

In the /system directory there are two other dictionaries related to the solution method. The first one is the fvSchemes dictionary that includes the entries for finite volume (fv) numerical schemes for every term, such as derivatives in equations, that are calculated during the simulation. This consists of 6 main sections: ddtSchemes (for first and second time derivatives \( \frac{\partial}{\partial t}, \frac{\partial^2}{\partial t^2} \)) gradSchemes (gradient \( \nabla \)), divSchemes (divergence \( \nabla \cdot \)), laplacianSchemes (the Laplacian \( \nabla^2 \)), interpolation schemes (for terms that are interpolations of values from cell centers to face centers), snGradSchemes (component of gradient normal to a cell face).

Figure 6.1 shows our entries for the steady-flow problem. Starting with the time schemes, we either choose the Euler discretization scheme, which is for transient flows, first order implicit and bounded, or the steadyState option which refers to steady-state flows, setting the time derivatives to zeros. There are also other types of discretization schemes, such as Crank Nicolson and backward schemes.

The second subcategory refers to gradient schemes. The default discretization scheme is Gauss linear. The Gauss term specifies the standard finite volume discretization of Gaussian integration, which requires the interpolation of values from the centers of cells to the centers of faces. The
interpolation scheme is given by the linear entry, meaning that the interpolation method used is the linear one. For our case we used the entry “cellLimited Gaus linear” for the velocity gradient, so that the face values do not fall outside the bounds of values in surrounding cells. This scheme is frequently used and included in a variety of tutorials. Furthermore, in the divergence schemes subcategory entries for both advection terms and diffusion terms are specified. Relying on similar problems to our, we specify those terms for velocity and turbulent kinetic energy as shown in Figure 6.1.

Finally, regarding the surface normal gradient schemes, we implement the following entry: limited corrected 0.33. A surface normal gradient is assessed at a cell face. It is actually the component of the gradient of values at the centers of the 2 cells that the face connects. As our geometry is kind of non-orthogonal, although we have set the limit to 65, we can add the correction factor 0.33 which increases the stability, but lacks accuracy, compared to 0.5 (this factor is known as ψ). To conclude, in the Laplacian schemes subcategory, the required entries are the gauss integration scheme, the interpolation scheme (linear) and the surface normal gradient scheme specified as previously.

![Figure 6.1. fvSchemes dictionary for specifying numerical schemes](image)

The second dictionary we are going to get a look into is the fvSolutionDict. Through this dictionary the equation solvers, tolerances and algorithms are controlled. Figure 6.2 represents our settings for a steady flow simulation using simpleFoam. The first entry of each section (velocity, pressure, etc.) refers to the solver used. This solver should not be confused with the application solver, such as simpleFoam and pisoFoam. The former are linear solvers for matrix equations and they include the following: PCG (Preconditioned Conjugate Gradient), smoothSolver (smoother is used), GAMG (generalized geometric-algebraic multi-grid), diagonal (diagonal solver for explicit systems). As shown in Figure 6.2 we use GAMG for the pressure field and SmoothSolver for the rest of the fields. In addition to the solvers, we should any additional preconditioners (e.g., DIC/DILU) and smoothers (e.g., GaussSeidel, symGaussSeidel).
The next entry that should be mentioned is tolerance. Tolerance is usually related to the initial residuals which are produced during the iterative calculation of the flow fields, and it actually represents the numerical error in the solution. The smaller this is, the more accurate the solution is. The solver stops the iterations as long as one of the following criteria are reached: 1) the initial residual is less than that specified in tolerance entry, 2) the ratio of current to initial residual is less than this specified in relTol entry, 3) the number of iterations surpasses the maximum number of Iterations specified by the user (maxIter).

Another important entry has to do with the relaxation factors mostly used in steady-state simulations (with simpleFoam). Under-relaxation factors are used in order to improve the stability of a computation, by limiting the amount a flow variable changes from one iteration to the next. Relaxation factors usually range from 0 to 1, however over-relaxation may be used as well in order to accelerate the convergence, but stability problems occur. Therefore, it is clear that, a proper choice of this factor is important for both convergence and stability of the computation. For our case after a number of trials, we concluded that when using simpleFoam, a relaxation factor close to 0.1 should be implemented. Figure 6.3 shows the differences in the initial residuals when using 2 different relaxation factors. It is clearly seen that a large relaxation factor for velocity, leads to an immediate divergence. Therefore, a trial-and-error method should be applied in order to set the appropriate relaxation factors for each flow field of our problem.

The rest of the options that can be noticed in Figure 6.2 in the pressure section, are related to the properties of the GAMG solver. In the end of the dictionary, some parameters regarding the algorithm of the solvers used are specified. If SIMPLE algorithm is executed, then the only parameter required is the one named nNonOrthogonalCorrectors and it defines repeated solutions.
of the pressure equations. Typically, this is set to 0 for steady-state or 1. In case of using PISO algorithm we should also specify the nCorrectors parameter which is the number of times the pressure equation is solved by the algorithm. We most commonly use 2 or 3.

6.2 Mesh Independence Study

After a complete description of the dictionaries required for running a case either in steady-state or in transient flow conditions, we can now proceed with the conduction of the mesh independence study. As already mentioned, this will be done by running the SimpleFoam solver to achieve a less time-consuming procedure. Consequently, a time-averaged flow will be introduced in the inlets of our geometry. This mesh independence study will be divided in two parts. The first part will involve 3 different mesh sizes (around 300k, 2.5m, 6m), and a general comparison of the flow field between the 3 meshes will be conducted. After selecting an appropriate size that is capable of a detailed detection of the flow field, we will advance with the second part of the study, that will help us choose the appropriate number of boundary layers, comparing the WSS values at a specific region of the geometry.

6.2.1 Turbulence model selection

It is of critical importance to mention that, all the simulations will be conducted using the LES (Large Eddy Simulation) model Dynamic k eqn. Although it is suggested that the flow within the LA is laminar and not significant turbulent fluctuations are observed [49], we must be certain that any transitional flows will be detected, therefore a turbulence model should be acquired. There is a great deal of models, yet they are all included in 3 major categories: DNS (Direct Numerical Simulation), LES and RANS (Reynolds Averaged Navier Stokes) models.

- DNS models solve directly the equations governing the fluid flow (the N-S), without using any modelling assumption, leading to very tiny time steps and detailed meshes. This means that the time and cost required are sky-rocketed, so DNS simulations are mostly used for academic purposes [81].
- LES models filter out spatially the smallest scales of turbulence, while the largest (those full of energy) scales are resolved directly. Since at a very small scales, the flow structures...
tend to be similar, simpler turbulence models are used. However, this still comes with an increased computational cost, though not as high as it happens with DNS.

- RANS models are time-averaged equations of fluid motion. These are similar to the original equation, but they include additional terms in the momentum equation known as Reynolds stress terms, that must be modelled since they are unknown. RANS models are the most widely used in industrial applications where extreme precision is not greatly required. From all 3 mentioned models, RANS is the least accurate but simultaneously the most time efficient.

As mentioned, for our mesh sensitivity analysis the LES model dynamic k eqn will be used, which is quite simple and not many entries are required to be set in the turbulence properties dictionary located in the /constant directory. Staying in the same directory, we can also specify the blood viscosity in the transport properties dictionary. Blood is assumed to be an incompressible Newtonian fluid with density $\rho = 1050 \text{ kg/m}^3$ and dynamic viscosity $\mu$ of $3.5 \times 10^{-3} \text{ Pa\cdot s}$ (and thus kinematic viscosity $\nu = 3.3 \times 10^{-6} \text{ m}^2/\text{s}$, which is defined in the dictionary), since the non-Newtonian effects of blood are negligible in large domains, such as the LA [83].

Taking a look into the source file of our turbulence model, we receive the following data (Figure 6.5) which correspond to Eq.(6.1) for the turbulence kinetic energy $k$ ($\text{m}^2\text{s}^{-2}$).

\[ \frac{D}{Dt} (\rho k) = \nabla \cdot (\rho D_k \nabla k) + \rho G - \frac{2}{3} \rho k \nabla \cdot u - \frac{C_e \rho k^{1.5}}{\Delta} + S_k , \]  

(6.1)

where $C_e$ and $C_k$ coefficients are derived from local flow properties, $S$ is a source term, $D_k$ is effective diffusivity for $k$, and finally $G$ accounts for the turbulent kinetic energy production due to the anisotropic part of the Reynolds-stress tensor ($\text{m}^2\text{s}^{-3}$).
Now, regarding the boundary conditions, before we proceed with analyzing those concerning pressure and velocity, we should mention what other flow fields are required for running an LES case. One is “nut” which corresponds to the turbulent (eddy) kinematic viscosity, and it is calculated during the simulation with an initial value of 0. The second and most critical one is the turbulent kinetic energy, and a fixed value as an initial condition is required for every patch. Either by defining the inlets or the outlet and setting to zero gradient the unknown we should take into account the Eq. (6.2) which provides the calculation of k for a smooth duct.

\[ k = \frac{3}{2}(U I)^2, \quad (6.2) \]

where U is the average velocity, and I is the turbulence intensity calculated as shown in Eq. (6.3).

\[ I = 0.16 R e^{-\frac{1}{8}}, \quad (6.3) \]

where Re is the Reynolds number based on the pipe hydraulic diameter d_h.

Executing simple calculations, since the flow is laminar, the k receives only very small values, of the order of 10 in the minus fourth.

Except for the above, velocity and pressure boundary conditions must also be set. Starting off with the velocity we will impose the velocity in the mitral valve patch derived from [84], which indicates both conditions, normal and AFib. Figure 6.6 shows the differences between normal “sinus” rhythm and Atrial Fibrillation in the mitral valve velocity change over time. It is clear that in the AFib case, the velocity curve lacks a second peak which is known as “A wave” which refers to the atrial contraction. As mentioned in the first chapter, atrial contraction which is responsible for approximately the 20 % left ventricular filling, is completely absent. However, the first peak, called “E wave” remains unchanged, since it does not depend on the atrial contraction.

In addition, since we are going to conduct the sensitivity analysis under steady-state condition we will use the average velocity (approximately 0.05 m/s) under AFib and impose it with a uniform
profile to the mitral valve patch. To do so, we take the advantage of “surfaceNormalFixedValue” boundary condition type which imposes the uniform profile normal to the patch surface. As for the rest, obviously the wall BC is set to no slip, while for the inlets a “pressureInletOutletVelocity” is implemented, which allows backflow, which is possible to occur since no valves are present in pulmonary veins. Concerning the pressure boundary conditions, a fixed value of zero is set for the inlets and a zeroGradient for the MV and the LA wall.

6.2.3 First Case Study

After defining the boundary conditions, we are now able to conduct the mesh independence study. First, we will compare three different mesh sizes relying on how efficiently they capture the blood flow with the LA.

The first one is generated by the Ansys Fluent Meshing tool as described in subchapter 5.2.5. We will run enough iterations to ensure that the model has converged to a solution. This mesh consists of around 300000 cells. The other two meshes are generated by snappyHexMesh. The second model contains far more cells, including 4 boundary layers. The number of cells increases dramatically to approximately 2.7 million. Finally, the third consists of roughly 8 million cells and a transient simulation for this mesh is out of our operational capabilities. However, we will conduct a steady-state simulation in order to demonstrate any differences compared to the second mesh.

Figure 6.6. Mitral valve velocity in normal rhythm and AFib. A wave is not present in AFib, indicating absence of atrial contraction.

![Mitral Velocity in Normal Rhythm](image1)

![Mitral Velocity in Atrial Fibrillation](image2)
After conducting the 3 mentioned simulations, we achieve the following results presented in Table 6.1

<table>
<thead>
<tr>
<th>Mesh</th>
<th>Number of Cells</th>
<th>Capturing the flow efficiently?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Fluent)</td>
<td>328885</td>
<td>No</td>
</tr>
<tr>
<td>2 (SnappyHexMesh)</td>
<td>2717786</td>
<td>Yes</td>
</tr>
<tr>
<td>3 (SnappyHexMesh)</td>
<td>8135720</td>
<td>Yes</td>
</tr>
</tbody>
</table>

*Table 6.1. 1st part of mesh independence study*

It is clear from the results that the first mesh, generated by Fluent, is incapable of capturing the flow as the other two do. After running for 30 thousand iterations the model indicated a backflow for the three out of 4 inlets, which is quite unphysical for our case. Still, unless we don’t increase the number of cells, we are not certain about the accuracy of the result. In addition, the velocity profiles of our inlets, were in most cases not parabolical. However, after running the second case for 30 thousand iterations, the results looked more stable and logical compared to the first simulation. A backflow in the RIPV inlet is observed, while for the three other inlets a perfect parabolic velocity profile is presented. We are therefore certain that the results are correct. In order to check how precise they are, we carry out the third simulation, with a mesh size of 8 million cells. In accordance with the results from the second study, we achieve the same picture for the third study as well. The third mesh also captures the backflow occurring in the RIPV inlet and the same exact parabolic velocity profiles in the rest of the inlets. As a consequence, since we cannot use this mesh in a transient simulation due to the huge computational cost, the second mesh can suit our requirements. Figures 6.8 to 6.10 show the differences in the velocity profiles between the three simulations. We can also examine the conditions inside the LA. Figure 6.7 shows the flow conditions (velocity) in a cross section of the LA for both the second and third meshes. The results are almost identical, confirming our decision to select the second mesh for proceeding with the transient simulation. Slight differences are due to the fact that we cannot capture the exact iteration under which they look the same. This snapshot corresponds to the time during which we detect the backflow in the RIPV inlet.

*Figure 6.7. Flow conditions in a cross section of meshes 2 and 3*
Figure 6.8. LIPV and LSPV inlets’ profile using mesh 1

Figure 6.9. LIPV and LSPV inlets’ profile using mesh 2

Figure 6.10. LIPV and LSPV inlets’ profile using mesh 3
6.2.4 Second Case Study

After selecting the approximate number of cells that we are going to use, we need to specify the number of layers that are required for a precise simulation. It is of great importance to determine that sine, our main goal is to calculate WSS indices, hence the flow close the walls must be efficiently captured. First, we are going to compare the differences in wall shear stress values in a specific region of the LA wall, acquiring from one to five boundary layers. If we are not satisfied with the absolute error produced comparing these five meshes, we can later add more layers. Table 6.2 gives data on the max WSS value each model captured (in the region depicted in Figure 6.11) under the same boundary conditions proposed in the previous case study.

<table>
<thead>
<tr>
<th>Boundary Layers</th>
<th>Number of Cells</th>
<th>Max WSS (Pa)</th>
<th>Absolute Error %</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2161911</td>
<td>4</td>
<td>11.11</td>
</tr>
<tr>
<td>2</td>
<td>2355792</td>
<td>3.7</td>
<td>2.78</td>
</tr>
<tr>
<td>3</td>
<td>2542349</td>
<td>3.2</td>
<td>11.11</td>
</tr>
<tr>
<td>4</td>
<td>2717786</td>
<td>3.6</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>2865591</td>
<td>3.6</td>
<td>-</td>
</tr>
</tbody>
</table>

*Table 6.2. Specifying the appropriate number of elements*

![Mesh Independence Study based on number of b layers](image)

*Figure 6.11. Bar graph of WSS values versus number of boundary layers*

It is clear from the above that since the absolute error between four and five boundary layers is exactly zero, we are going to use the mesh with the least cells. As a general trend, we can see that there are no big differences between the models, and even the first model can capture almost efficiently the WSS values. There is a slight fluctuation, however, which results from the third mesh that underestimates the WSS values. This probably has to do with either some oscillations of the residuals after the model has converged to a solution or a not perfect layer addition close to the wall. Figure 6.12 illustrates the distribution of WSS among all five models. No big differences are observed in general, whereas the fourth and fifth model provide a smoother distribution. We still need to mention that, for our study, not a high accuracy is required in terms of WSS values, as what we aim to capture is the distribution, which is correctly done by all five models. We can
still interpret the results, as the high WSS value in the region depicted in Figure 6.12 is due to the convergence of the flow coming from the inlets and finally hitting the wall of the LA chamber. Some regions also seem to have increased WSS values, but they are insignificant compared to the main region of interest. Finally, if we aimed to have a great accuracy as well in the values and not only in the distribution, we would definitely choose the mesh consisting of 8 million cells, but this is not the case of our study. At that point, it’s important to mention, that the LES model provides the same exact results as running a simple laminar model. Therefore, we are certain that the flow under such conditions is laminar.

6.2.5 Effect of inaccuracies in mitral flow on the WSS distribution
Aiming to ensure that inaccuracies in the mitral boundary condition implemented will not greatly affect the WSS values and most significantly the distribution, it would be appropriate to conduct a study in which we will compare the final WSS distribution after changing the mitral velocity, for instance ± 20% compared to the initial one. We demonstrate the results in Table 6.3.

<table>
<thead>
<tr>
<th>Mitral Valve Velocity (m/s)</th>
<th>Max WSS (Pa)</th>
<th>Difference (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.04 (-20%)</td>
<td>2.4</td>
<td>33.33</td>
</tr>
<tr>
<td>0.05</td>
<td>3.6</td>
<td>-</td>
</tr>
<tr>
<td>0.06 (+20%)</td>
<td>4.8</td>
<td>33.33</td>
</tr>
</tbody>
</table>

*Table 6.3. Estimation of inaccuracies in WSS distribution*

After increasing and decreasing the MV velocity equally by 20%, we notice a 33% difference in the maximal WSS value, which is quite reasonable since WSS is directly affected by velocity. However, no significant changes in the distribution are observed. The high and the almost zero WSS value regions remain unchanged.
6.2.6 Comparison between LES, k-Epsilon, k-OmegaSST turbulence models

In our final test case, we are going to compare any differences between the LES model and the RANS models k-Epsilon and k-OmegaSST. In addition, we are going to implement the second type of boundary conditions, for which the parabolic velocity profile is manually implemented on the inlet patches, and an “inletOutlet” velocity BC which prevents backflow (in contrast to the pulmonary veins where a valve is not present), is acquired for the MV patch.

As mentioned, the parabolic profile will be implemented manually to each inlet, as an option for that BC is not included in OpenFOAM. In order to do that, we use the codedFixedValue type of BC. What we have to do is to define the normal vector \( \hat{n} \) for each inlet, and then for each face that is contained in the defined patch we simply define the velocity profile \( V(x,y,z) \) as described in Eq.(6.4) which corresponds to Poiseuille law in three dimensions

\[
V(x,y,z) = 2u_{avg} \left( 1 - \frac{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}{R^2} \right),
\]

where, \( u_{avg} \) is the average velocity, \( (x,y,z) \) are the coordinates of each face centroid, and \( (x_0, y_0, z_0) \), are the coordinates of the patch center.

The boundary limits of each patch can be found through ParaView, which actually defines a box that contains the whole patch, mentioning the boundaries for each of the three dimensions. To find its center, we simply find the average value and then we can add these values as \( (x_0, y_0, z_0) \), in our /U case file. Since we are given only (by Geneva University Hospital) the flowrate for one pulmonary vein, we are going to assume that this applies for every PV regardless of their diameter. After averaging the flow rate in time (since we are implementing steady-state conditions at the moment) we get a value of \( Q_{\text{average}} = 1.5794 \text{ m}^3/\text{s}. \) As we are going to implement the parabolic profile described in Eq.(6.4) we only need to specify the average velocity for each inlet, and the inlet radius (diameter and consequently area). The latter is extracted by ParaView, and the former

Figure 6.13. WSS distribution comparison between -20% and +20% difference in MV velocity
results from the flowrate definition as flowrate divided by the cross-area of the patch. Thus, we receive Table 6.4 which presents the necessary data for each inlet.

<table>
<thead>
<tr>
<th>Inlet</th>
<th>Area $(m^2)$</th>
<th>Diameter (m)</th>
<th>Flow rate $(m^3/s)$</th>
<th>Average velocity $(m/s)$</th>
<th>Reynolds Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSPV</td>
<td>0.00017</td>
<td>0.01493</td>
<td>1.58E-05</td>
<td>0.09027</td>
<td>408.2776</td>
</tr>
<tr>
<td>RIPV</td>
<td>9.6E-05</td>
<td>0.01106</td>
<td>1.58E-05</td>
<td>0.16447</td>
<td>551.0987</td>
</tr>
<tr>
<td>LSPV</td>
<td>9.92E-05</td>
<td>0.01124</td>
<td>1.58E-05</td>
<td>0.15923</td>
<td>542.2524</td>
</tr>
<tr>
<td>LIPV</td>
<td>4.14E-05</td>
<td>0.00726</td>
<td>1.58E-05</td>
<td>0.38189</td>
<td>839.7667</td>
</tr>
</tbody>
</table>

Table 6.4. Area, Diameter, Flow rate, velocity, and Reynolds number for every inlet

In Table 6.4, we also included the Reynolds number in order to demonstrate that blood, at least in the inlet region is flowing under laminar conditions (even when considering the maximal flow rate). Consequently, as every parameter is now specified, we can proceed with the simulation results.

It is important to know that for the k-epsilon model apart from initializing a value for the turbulent kinetic energy $k$, we should also acquire a new parameter known as turbulence dissipation rate $\varepsilon$, which is given in Eq.(6.5):  
$$\varepsilon = \frac{C_m^{0.75}k^{1.5}}{L},$$  
(6.5)

where $C_m$ is a model constant equal to 0.09 by default and $L$, is a reference length scale, for our case the diameter (m).

As Grigoriadis et al [85] proposed, k-Epsilon model offers the greatest stability among all models, when computing the flow in the left atrium. However, this is partly correct to apply for a case like that, since using a RANS model completely reduces the order of accuracy, averaging the flow field, and introducing an additional dissipation, which results in a better stability and convergence. This will be obviously proved in the following parts.

In addition, in order to run the k-omega Shear Stress Transport model, apart from the turbulence kinetic energy constrain, we should specify the turbulence specific dissipation rate as well, as follows:

$$\omega = \frac{k^{0.5}}{C_\mu^{0.25}L},$$  
(6.6)

K-omega SST model is suited for simulation flow near the walls, and it is a hybrid model combining both k-omega and k-epsilon models. First of all we are going to compare the initials residuals of velocity (in x direction) between these three models (LES, k-epsilon, and k-Omega-SST). Results are shown in Figure 6.14.
We can clearly notice the influence of the additional dissipation added by the RANS models, compared to the LES model. The initial residuals of \( U_x \) remain almost the same after 2000 iterations concerning the LES model, whereas for the other two models, they are observed to continue decreasing moderately. However, regarding the K-Epsilon model, it is quite clear how faster it converges compared to the other two. It obviously demonstrates a better “behavior”, however we should investigate whether its WSS distribution is in accordance with what the LES and k-OmegaSST computed.

Before advancing, we can take a look into the implemented parabolic profile to the inlets. Figure 5.15 shows the LIPV patch velocity profile, while Figure 6.16 Illustrates the flow field near the inlets LIPV and LSPV. Since the LIPV patch has the smallest area, it is also going to have the highest velocity compared to the other three inlets.
We can then continue by comparing the WSS distributions between the three models. Regarding the WSS distribution in general, we can see that a specific region experiences high levels of WSS, as a result of the converging-like flow field which is created by the high blood velocity initiating from the LIPV inlet. The distribution shown in Figure 6.17 Corresponds to the LES model, whereas Figure 6.18 and 6.19 to k-OmegaSST and k-Epsilon respectively.
In general, all models capture increased WSS values in the same regions, however it is observed that only the LES and the k-OmegaSST ones detect the highest value in the region opposed to the LIPV inlet, which is in fact the most logical result. In contrast, the k-Epsilon model detects the highest value in a region close to the LIPV inlet, though this is not a region of interest, since it is manually added using the VMTK software in order to create a fully circular inlet patch.

![Figure 6.18. WSS distribution using the k-OmegaSST model](image1)

![Figure 6.19. WSS distribution using the k-Epsilon model](image2)
Concerning the first two models, the distributions are almost identical, and only a difference of 1.1 Pa in WSS is observed. Nevertheless, as discussed previously, we will still utilize the LES model for the transient simulation since we have the computational ability to do so. As shown in Figure

6.3 Transient Flow Simulation

For the conduction of our transient flow simulation, we must decide what type of boundary conditions will be utilized. We have already discussed about two types, one for which the MV velocity is known, and the other for which the PV flow rate is known for one of our inlets. However, the latter does not take into consideration the closure of the mitral valve during the cardiac cycle and considers a constant flow through it. Therefore, in order to carry out a study as realistic as possible we will use the first type of BCs, which are presented in Table 6.5. The mitral valve velocity is as said depicted in Figure 6.6, where the closure of the valve and a small regurgitation take place.

<table>
<thead>
<tr>
<th>Patch</th>
<th>Boundary Condition</th>
<th>U</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mitral Valve</td>
<td>Dirichlet (Uniform Profile – u(t))</td>
<td>( \nabla p = 0 )</td>
<td></td>
</tr>
<tr>
<td>LIPV</td>
<td>Mixed ( \frac{\partial u}{\partial n} = 0 ) / Dirichlet (allows backflow)</td>
<td>( p=0 )</td>
<td></td>
</tr>
<tr>
<td>LSPV</td>
<td>Mixed ( \frac{\partial u}{\partial n} = 0 ) / Dirichlet (allows backflow)</td>
<td>( p=0 )</td>
<td></td>
</tr>
<tr>
<td>RIPV</td>
<td>Mixed ( \frac{\partial u}{\partial n} = 0 ) / Dirichlet (allows backflow)</td>
<td>( p=0 )</td>
<td></td>
</tr>
<tr>
<td>RSPV</td>
<td>Mixed ( \frac{\partial u}{\partial n} = 0 ) / Dirichlet (allows backflow)</td>
<td>( p=0 )</td>
<td></td>
</tr>
<tr>
<td>Wall</td>
<td>No slip / rigid wall</td>
<td>( \nabla p = 0 )</td>
<td></td>
</tr>
</tbody>
</table>

*Table 6.5. Boundary Conditions for transient flow study*

Before initiating the simulation, for which we will perform the pisoFoam solver, we make sure that we use the appropriate boundary condition for the outlet patch (MV). In this case, we need to take advantage of the codedFixedValue boundary condition provided by OpenFoam, since we should create a temporally varying profile based on a velocity-time data table. We implement this uniform profile on MV patch by using linear interpolation for our velocity data in order to have a smooth behavior and steer clear of sharp changes as our data might indicate (since we only have information about some time points throughout the cardiac cycle of 0.8 sec). The required code is presented in Appendix III.

Moreover, we must specify some parameters included in controlDict dictionary which located in /system directory. The most important one is the simulation time step. This is defined based on the value of CFL number which should be lower than one for a stable simulation. For our type of mesh which almost consists of 3 million cells, and a maximal velocity of around 0.3m/s, we would set our time step to \( 5 \cdot 10^{-6} \) sec, which is quite small and may lead to very long simulation intervals. However, we know in advance that such types of simulations take up to approximately 10 days
and more so we set an adjustable time step which constantly leads to a CFL number lower than 1 in order to ensure that our simulation will remain stable for the whole procedure. In this way, we reduce the total simulation time by at least 70%, which gives us the opportunity to run more than one cardiac cycle, avoiding the effects of transitional flow. It is important to mention that this capability of using an adjustable time step is not provided by the pisoFoam solver. There are two ways this can be done. The first is to modify the source code of pisoFoam solver with the addition of two header files, as shown in Appendix I, and then set the required Courant number limit in the controlDict Dictionary. The second and more simple way is to run the pimpleFoam solver for which this capability is built in. After setting nOuterCorrectors to 1, pimpleFoam is going to run essentially as pisoFoam.

In addition, we should specify the number of times we would like to save our results (writeInterval option) during the simulation. The final WSS indices and flow fields will be based on the saved results so it’s important to capture at least the most significant time points of the 0.8 sec simulation procedure. Finally, in the controlDict dictionary, we should also specify a function, which is going to detect the vortex structures in our flow domain. This is done by either q criterion or lambda2 criterion function objects. A snapshot of Lambda2 criterion source file is presented in Figure 6.20. The equation depicted actually corresponds to the already given in Subchapter 3.1.7.

Simulation for one cardiac cycle lasted approximately 170000 seconds or almost two days, due to the utilization of an adjustable time step. Results on flow patterns, WSS distributions, vortex formation, TAWSS, ECAP and RRT indices are presented in the following chapter.

![Lambda2 criterion source file](image)
7 Results

Figure 7.1. LIPV and LSPV inlets’ velocity profile under maximal atrial contraction

Figure 7.2. LIPV and LSPV inlets’ velocity profile under MV reverse flow
**Figure 7.3.** Blood flow patterns under MV reverse flow

**Figure 7.4.** Blood flow patterns under maximal atrial contraction
**Figure 7.5.** Vortex structures detected under maximal atrial contraction using lambda2 criterion

**Figure 7.6.** Vortex structures detected under MV reverse flow using lambda2 criterion
Looking at Figures 7.1 to 7.4, it is clear that during the maximal atrial contraction we observe the highest velocity values while during the small mitral backflow, a backflow in the pulmonary veins is also detected, which is reasonable due to the boundary conditions we imposed, as the pulmonary veins do not contain any valves. However, we should mention that under real cardiac conditions, the backflow in the mitral valve and in the pulmonary veins does not occur simultaneously. Nevertheless, this does not mean that our results are not correct, as this happens due to the fact that we have neglected the wall motion, since even in terms of FSI modeling we wouldn’t be able to simulate the synchronous and complicated wall motion. The best method to implement will be presented later in the conclusion chapter.

Regarding the flow patterns, both under the conditions of atrial contraction and reverse flow, we cannot observe any noticeable differences apart from the velocity values. During atrial contraction the maximal velocity is 1.2 m/s while during the mitral backflow this drops to 0.26 m/s. In addition, the Left Atrial Appendage experiences very low values of velocity which as we will later mention, will lead to extremely low WSS values. As for the vortex structures formation in the left atrium, it is clear that when the valve is open there are more vortices that when the valve is closed (or in backflow). It is quite interesting to mention, that mostly during the atrial contraction, we can notice the formation of vortex rings in the regions close to the pulmonary veins. Vortex rings can also form after blood exits the mitral valve towards the left ventricle.

Figures 7.7 and 7.8 illustrate the Wall Shear Stress distribution on the LA wall under MV reverse flow and maximal atrial contraction. It is obvious that when the mitral valve is almost closed or under backflow wall shear stress is still proportional to the velocity, therefore there is no doubt that during such conditions, WSS values will be extremely low. The only regions we can observe values close to 1 Pa are those close to the inlets. There is also a region of around 0.2 Pa close to the mitral valve, which is reasonable since the flow is heading towards the LA. As for the rest of the LA wall, values are mostly under 0.2 Pa, while some regions experience almost negligible shear stresses. One of these regions is the LAA which due to its morphology is a high-risk region for blood stasis and thrombus formation. After analyzing the mentioned indices in specific time points during the cardiac cycle, we can proceed with the investigation of time averaged indices.
Figure 7.9 depicts the Time Averaged Wall Shear Stress distribution calculated using the Eq. (3.40). For the computation of this parameter in OpenFOAM, we take advantage of the “temporal statistics” filter in ParaView. This feature is capable of calculating the average value of every magnitude, including velocity, pressure, WSS etc. For this reason, before we actually use this filter, we compute the absolute values of WSS in every direction, so we have the following indices: $|WSS_x|$, $|WSS_y|$, $|WSS_z|$. As long as these magnitudes are calculated, we use the mentioned filter in order to compute their average values and then TAWSS is determined as shown in Eq. (7.1):

$$TAWSS = \sqrt{|WSS_x|_{avg}^2 + |WSS_y|_{avg}^2 + |WSS_z|_{avg}^2},$$  

(7.1)

where, $|WSS_{x,y,z}|_{avg}$ is the average value of the absolute wall shear stress value in the $x,y,z$ direction for a cardiac cycle.
Once again, we can observe that the regions close to the inlets experience the highest WSS values. The area where the flow from the two inlets (LIPV and LSPV) converge, also experiences WSS values over 2 Pa on average. Furthermore, it seems that we get a similar picture of the distribution with the results from implementing velocity boundary conditions both in the inlets and the outlet (see Figures 6.12 and 6.17). In general, most of the LA wall is exposed to WSS values of approximately 1 Pa, whereas the WSS in the LAA region is extremely low, in accordance with related studies [49, 85, 86].

Figure 7.10 illustrates the Oscillatory Shear Index distribution on the LA wall calculated as shown in Eq.(3.41). It’s important to mention once again that the OSI index indicates the change of direction of the WSS vector during the cardiac cycle. For example, the OSI is high in regions where the WSS changes much in the specified time interval. In our case, we can notice that the regions that are exposed to relatively high TAWSS actually lead to low OSI values. It’s interesting to mention that in the LAA, OSI seems to be increased, despite the fact that the WSS values are almost negligible.

Endothelial Cell Activation Potential distribution on the LA wall is presented in Figure 7.11. As mentioned, ECAP regions with high OSI and low TAWSS values (see Eq.(3.42)). We can clearly observe that the only region with increased ECAP values, is the LAA, confirming the location of high thrombosis risk. The deeper we move into the LAA region, the highest the value is. A peak value of 10 Pa$^{-1}$ is observed, which is in agreement with the results from studies focusing on the role of the LAA in thrombus formation [49, 87, 88].

Finally, the RRT index combines both the OSI and TAWSS indices (Eq.(3.43)), as it captures the residence time of blood particles close to the atrial wall. In accordance with the ECAP index distribution, RRT increases the closer we head into the LAA, indicating the low velocity of blood. A peak value of 200 Pa$^{-1}$ can be noted in the LAA tip, which is compliant with related studies.
Figure 7.11. Endothelial Cell Activation Potential distribution

Figure 7.12. Relative Residence Time distribution
8 Conclusion

Coming to the end of the present Diploma Thesis, it can be said that it fulfills its purpose as a reliable tool for studying the blood flow within the Left Atrium, and most importantly, generating mappings of the numerous Wall Shear Stress indices that can be later compared to electrophysiological mappings using patient specific data. The framework that has been developed can be easily adapted to the requirements of every patient, in terms of 3D model optimization, mesh generation and implementation of the boundary conditions. The automation of this process could possibly provide medical experts with the required results in much less time, however we are quite far from it, as a lot of work must be done in the management of patient-specific data.

Summarizing the main topics of this thesis, first, we attempted to provide a thorough view of the medical background required concerning Atrial Fibrillation and generally the function and anatomy of the heart. Afterwards, we aimed to present the close connection between Fluid Mechanics and Hemodynamics, introducing key parameters and indices that would be later calculated computationally. Furthermore, since the current thesis focuses on the computational part of the project we participate in, an in-depth analysis on the fundamental numerical methods used, was conducted, as we then concentrated on a detailed investigation of the numerous capabilities the open-source software OpenFOAM® provides.

The preprocessing part included methods regarding the optimization of the three-dimensional model, as the latter results from the reconstruction through MRI images. This process is followed by the mesh generation procedure, which is of paramount importance, as a bad quality mesh will possibly lead to either instabilities during the solution or eventually the divergence of the solution. Mesh generation using the snappyHexMesh utility of OpenFOAM, is certainly not as easy as using commercial software like Ansys Fluent Mesher, since the learning curve is quite steep, and a lot of tries should be attempted before achieving a good quality. However, the user achieves a great level of understanding of several mesh topology parameters.

As for the solution method, we first conducted the mesh independence study under steady state conditions, implementing an average velocity to either the inlets or the outlet. Both boundary conditions had to be imposed manually using C programming language. The LES model did not present any noticeable differences compared to the laminar model, confirming our assumption about laminar flow conditions in the LA (as also presented in [49]). As for the rest of the turbulence models performed, kOmegaSST presented a similar picture of the WSS distribution to the one using the LES model, while k-Epsilon, although converging faster, was not so accurate compared to the other two. A case study conducted with future prospects of its consideration was to examine the inaccuracies of the mitral valve velocity on the Wall Shear Stress distribution. Results showed that the distribution remains almost unchanged, although the peak values change as the MV velocity does.

Regarding the transient flow simulation, the LES model was implemented, while the mitral valve velocity change over time was imposed as a boundary condition. The utilization of an adjustable time-step led to a drastic reduction of the simulation time (instead of defining a single one e.g. 5E-
6) to roughly 2 days. Results are consistent with those of related studies, and WSS distributions can now be compared to electrophysiological mappings.

Figure 8.1. TAWSS compared to voltage mapping of the LA [90]

8.1 Limitations
It is important to note that for the present study, we assumed that the LA wall is rigid, completely ignoring its dynamic movement during the cardiac cycle. The implementation of fluid structure interaction (FSI) techniques would still not solve this issue, as the physical structure of the heart chamber is a dynamic parameter, which makes it much more complicated to simulating throughout the cardiac cycle. Even a two-way simplified FSI solution would not be able to capture efficiently the flow, despite the fact that the effect of blood flow on the wall and respectively the wall’s effect on blood flow are taken into account. The reason is that the cardiac wall motion is not the result from the blood hitting the wall, but it depends on the contraction and relaxation of the heart due to its electrical activity. However, due to the abnormal atrial beating, studies [49] have shown that rigid wall simulations don’t differ a lot compared to those considering deformable walls.

In addition, another factor that we have not taken into account is the volume of our model. We do not have any data regarding the time point during the cardiac cycle that the images for the 3D model reconstruction were taken. Therefore, the LA volume could correspond to either systole or diastole or even an average value of those two. Finally, although we were capable of implementing parabolic velocity profile to the inlets, for our transient flow simulation, we imposed a uniform profile to the mitral valve patch, which can possibly affect the WSS distribution, and the flow patterns in general, while it neglects the motion of its leaflets.

8.2 Future work
A lot can be done in a direction of improving the outcomes of the current study. Regarding the implementation of a dynamic wall motion, several methods can be considered. The first is to consider a technique which does not take into account the interactions between the blood and the wall, but the solid-fluid interface has a prescribed velocity, equal to the fluid one with no slip conditions on the wall. The movement of the wall is then imposed, and it depends on the volume change over time. The complete procedure is described in [88]. Another way is to create a dynamic
mesh by combining 3D models of the LA captured in different time points during a cardiac cycle [49]. However, this method depends on the capabilities of the imaging technique used.

In addition, aiming to conduct a realistic as possible CFD study, we can improve the boundary conditions imposed to the inlets (pulmonary veins) or the outlet (mitral valve). Instead of implementing uniform velocity profiles, a method that has been developed for this study, but wasn’t utilized, is to convert phase contrast MRI images of the cross sections of our patches directly into temporally and spatially varying boundary conditions. Phase contrast MRI technique creates images that provide flow measurements of the captured area. This requires the development of a program that would firstly orient the 2D image properly in the 3D model and then match the pixels of it (each picture voxel or pixel has a different colour corresponding to a different velocity value) with the mesh cells’ centroids. In this way we allocate a velocity value to each centroid for a single time step only, and by using several images captured during the cardiac cycle, we are able to create the mentioned temporally and spatially varying boundary conditions. An example is shown in Figure 8.2.

![Figure 8.2. Acquisition of phase contrast MRI images as boundary conditions to the patches](image)

Finally, a study that would be quite interesting to conduct, is to design a conceptual left atrial appendage occlusion device, as we can investigate the changes in the WSS distribution and generally the flow patterns in the LA with the device and not. It is obvious that excluding the LAA region from the flow, would dramatically decrease the ECAP and RRT indices, as blood would not be able to pass through the device, as shown in Figure 2.19.
9 References


[47]. Schoenhagen, P., White, R. D., Nissen, S. E., & Tuzcu, E. M. (2003). Coronary imaging: Angiography shows the stenosis, but ivus, CT, and MRI show the plaque. Cleveland Clinic Journal of Medicine, 70(8), 713-719. doi:10.3949/ccjm.70.8.713


[55]. All there is to know about different mesh types in CFD! (2016, June 01). Retrieved June 1, 2022, from https://www.manchestercfd.co.uk/post/all-there-is-to-know-about-different-mesh-types-in-cfd


[70]. OpenFOAM. Retrieved June 1, 2022, from https://www.openfoam.com/

[71]. Holtzmann, T. Mathematics, Numerics, Derivations and OpenFOAM. *Holzmann CFD, Release 7.0*. doi:10.13140/RG.2.2.27193.36960

[72]. 3D slicer image computing platform. Retrieved June 1, 2022, from https://www.slicer.org/

[73]. Mathew, R. C., & Kramer, C. M. (2018). Recent advances in magnetic resonance imaging for peripheral artery disease. *Vascular Medicine, 23*(2), 143-152. doi:10.1177/1358863x18754694

[74]. Home of the blender project - free and open 3D creation software. Retrieved June 1, 2022, from https://www.blender.org/


References


Appendix I

Addition of Temperature field and adjustable time step

- myIcoFoam

```cpp
1. /*----------------------------------*|
2. \|    Field  OpenFOAM: The Open Source CFD Toolbox
3. \|   Operation Website: https://openfoam.org
4. \|   And  Copyright (C) 2011-2018 OpenFOAM Foundation
5. \|  Manipulation                                 |
6. *---------------------------------------------------------------------------*/
7. 
8. License
9. This file is part of OpenFOAM.
10. OpenFOAM is free software: you can redistribute it and/or modify it
11. under the terms of the GNU General Public License as published by
12. the Free Software Foundation, either version 3 of the License, or
13. (at your option) any later version.
14. OpenFOAM is distributed in the hope that it will be useful, but WITHOUT
15. ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or
16. FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License
17. for more details.
18. You should have received a copy of the GNU General Public License
19. along with OpenFOAM. If not, see <http://www.gnu.org/licenses/>.
20. 
21. Application
22. icoFoam
23. 
24. Description
25. Transient solver for incompressible, laminar flow of Newtonian fluids.
26. 
```
55. #include "CourantNo.H"
56. #include "setDeltaT.H"
57. // Momentum predictor
58. fvVectorMatrix UEqn
59. {
60.   fvm::ddt(U)
61.     + fvm::div(phi, U)
62.     - fvm::laplacian(nu, U)
63. }
64. if (piso.momentumPredictor())
65. {
66.   solve(UEqn == -fvc::grad(p));
67. }
68. // --- PISO loop
69. while (piso.correct())
70. {
71.   volScalarField rAU(1.0/UEqn.A());
72.   volVectorField HbyA(constrainHbyA(rAU*UEqn.H(), U, p));
73.   surfaceScalarField phiHbyA
74.   ("phiHbyA",
75.     fvc::flux(HbyA)
76.     + fvc::interpolate(rAU)*fvc::ddtCorr(U, phi)
77.   );
78.   adjustPhi(phiHbyA, U, p);
79.   // Update the pressure BCs to ensure flux consistency
80.   constrainPressure(p, U, phiHbyA, rAU);
81.   // Non-orthogonal pressure corrector loop
82.   while (piso.correctNonOrthogonal())
83.   {
84.     // Pressure corrector
85.     fvScalarMatrix pEqn
86.     {
87.       fvm::laplacian(rAU, p) == fvc::div(phiHbyA)
88.     }
89.     pEqn.setReference(pRefCell, pRefValue);
90.     pEqn.solve();
91.     if (piso.finalNonOrthogonalIter())
92.     {
93.       phi = phiHbyA - pEqn.flux();
94.     }
95.   }
96.   // Update the pressure
97.   constrainPressure(p, U, phiHbyA, rAU);
98.   U = HbyA - rAU*fvc::grad(p);
99.   U.correctBoundaryConditions();
100. }
101.  
102. 
103.
Appendix I

116. //add these lines...
117.   fvScalarMatrix TEqn
118.   {
119.       fvm::ddt(T)
120.       + fvm::div(phi, T)
121.       - fvm::laplacian(alpha, T)
122.   };
123.   TEqn.solve();
124.   //done adding lines...
125.
126.   runTime.write();
127.   Info<< "ExecutionTime = " << runTime.elapsedCpuTime() << " s"
128.       << " ClockTime = " << runTime.elapsedClockTime() << " s"
129.       << nl << endl;
130. }
131.
132.   Info<< "End\n" << endl;
133.   return 0;
134. }
135.
136.   // ************************************************************************
137.   // CreateFields
138.   // ************************************************************************
139.
140.   Info<< "Reading transportProperties\n" << endl;
141.   IOdictionary transportProperties
142.   {
143.       IOobject
144.         ("transportProperties",
145.          runTime.constant(),
146.          mesh,
147.          IOobject::MUST_READ_IF_MODIFIED,
148.          IOobject::NO_WRITE
149.         );
150.
151.   dimensionedScalar nu
152.   ( 153.       "nu",
154.       dimViscosity,
155.       transportProperties.lookup("nu")
156.   );
157.
158.   dimensionedScalar alpha
159.   ( 160.       transportProperties.lookup("alpha")
161.   );
162.   //Done for now...
163.
164.   Info<< "Reading field p\n" << endl;
165.   volScalarField p


31. ( 
32.   IOobject 
33.     (
34.       "p",
35.       runTime.timeName(),
36.       mesh,  
37.       IOobject::MUST_READ,
38.       IOobject::AUTO_WRITE
39.     ),
40.     mesh 
41. );
42. 
43. Info<< "Reading field U\n" << endl;
44. volVectorField U 
45. ( 
46.   IOobject 
47.     (  
48.       "U",
49.       runTime.timeName(),
50.       mesh,  
51.       IOobject::MUST_READ,
52.       IOobject::AUTO_WRITE
53.     ),
54.     mesh 
55. );
56. Info<< "Reading field T\n" << endl;
57. volScalarField T 
58. ( 
59.   IOobject 
60.     (  
61.       "T",
62.       runTime.timeName(),
63.       mesh,  
64.       IOobject::MUST_READ,
65.       IOobject::AUTO_WRITE
66.     ),
67.     mesh 
68. );
69. );
70. 
71. #include "createPhi.H"
72. 
73. label pRefCell = 0;
74. scalar pRefValue = 0.0;
75. setRefCell(p, mesh.solutionDict().subDict("PISO"), pRefCell, pRefValue);
76. mesh.setFluxRequired(p.name());
Appendix II

*VMTK commands*

- **Flow extensions**

```
1. vmtksurfacereader -ifile clippedLA_ext1.stl --pipe vmtkcenterlines -seedselector openprofiles --pipe vmtkflowextensions -adaptivelength 1 -extensionratio 20 -normalestimationratio 1 -interactive 1 --pipe vmtksurfacewriter -ofile clippedLA_ext2.stl
```

- **Surface clipping**

```
1. vmtksurfaceclipper -ifile geometry_from_blender.stl -ofile clippedLA.stl
```
Appendix III

- Velocity boundary conditions for transient flow study

1. dimensions [0 1 -1 0 0 0];
2. internalField uniform (0 0 0);
3. boundaryField {
4. RSPV {
5. type pressureInletOutletVelocity;
6. value uniform (0 0 0);
7. }
8. RIPV {
9. type pressureInletOutletVelocity;
10. value uniform (0 0 0);
11. }
12. LSPV {
13. type pressureInletOutletVelocity;
14. value uniform (0 0 0);
15. }
16. LIPV {
17. type pressureInletOutletVelocity;
18. value uniform (0 0 0);
19. }
20. Mitral {
21. type codedFixedValue;
22. value uniform (0 0 0);
23. redirectType inletcond;
24. code #{
25. const vectorField& Cf = patch().Cf();
26. vectorField& field = *this;
27. vectorField nHat = this->patch().nf();
28. scalar t = this->db().time().value();
29. const scalar T = 0.8;
30. int j;
31. forAll(Cf,faceI) {
32. const scalar x = Cf[faceI][0];
33. const scalar y = Cf[faceI][1];
34. const scalar z = Cf[faceI][2];
35. }
double step[82]={0,
0.02, 0.04, 0.06, 0.08, 0.1, 0.12, 0.14, 0.16, 0.18, 0.2, 0.22, 
0.24, 0.26, 0.28, 0.3, 0.32, 0.34, 0.36, 0.38, 0.4, 0.42, 0.44, 
0.46, 0.48, 0.5, 0.52, 0.54, 0.56, 0.58, 0.6, 0.62, 0.64, 0.66, 
0.68, 0.7, 0.72, 0.74, 0.76, 0.78, 0.8, 0, 0.02, 0.04, 0.06, 0.08, 
0.1, 0.12, 0.14, 0.16, 0.18, 0.2, 0.22, 0.24, 0.26, 0.28, 0.3, 
0.32, 0.34, 0.36, 0.38, 0.4, 0.42, 0.44, 0.46, 0.48, 0.5, 0.52, 
0.54, 0.56, 0.58, 0.6, 0.62, 0.64, 0.66, 0.68, 0.7, 0.72, 0.74, 
0.76, 0.78, 0.8};

double Uavg[82]={0,
-0.05, -0.08, -0.06, -0.03, -0.02, -0.01, -0.005, -0.0045, -0.004, 
-0.0035, -0.003, -0.0025, -0.002, -0.0015, -0.001, 
0, 0, 0, 0, 0.03, 0.07, 0.15, 0.25, 0.31, 0.32, 0.31, 0.25, 0.2, 
0.15, 0.1, 0.07, 0.05, 0.02, 0.005, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 
-0.05, -0.08, -0.06, -0.03, -0.02, -0.01, -0.005, -0.004, -0.0035, -0.0025, 
-0.002, -0.0015, -0.001, 
0, 0, 0, 0, 0.03, 0.07, 0.15, 0.25, 0.31, 0.32, 0.31, 0.25, 0.2, 
0.15, 0.1, 0.07, 0.05, 0.02, 0.005, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0};

for (j=0;j<82;j++)
{
    if ((fmod(t,T) >= step[j]) && (fmod(t,T) < step[j+1]))
    {
        field[faceI]=((fmod(t,T)-step[j])/(step[j+1]-step[j]))*((nHat[faceI])*Uavg[j+1]-
                       (nHat[faceI]*Uavg[j])) +nHat[faceI]*Uavg[j];
    }
}

#include "fvCFD.H"
#include <cmath>
#include <iostream>
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <math.h>


type noSlip;
Velocity boundary condition for mesh independence study (RSPV inlet only)

```
1. RSPV
2. {
3.    type codedFixedValue;
4.    value uniform (0 0 0);
5.    redirectType inletCond;
6. }
7. code
8. #{
9.    const vectorField& Cf = patch().Cf();
10.   vectorField& field = *this;
11.   scalar Uavg = -0.09027;
12.   vectorField nHat = this->patch().nf();
13.   scalar R=0.007449;
14.   vector Uinl;
15.   forAll(Cf,faceI)
16.     {
17.       const scalar x = Cf[faceI][0];
18.       const scalar y = Cf[faceI][1];
19.       const scalar z = Cf[faceI][2];
20.       //Info<<nHat<<endl;
21.       field[faceI]=(nHat[faceI]*(2*Uavg*(1-(pow(x+0.03225,2) + pow(y-0.056875,2) + pow(z+0.0208,2))/pow(R,2))));
22.     }
23.   //Info<<nHat<<endl;
24. }
25. #};
26. codeOptions
27. #{
28.    -I$(LIB_SRC)/finiteVolume/lnInclude \
29.    -I$(LIB_SRC)/meshTools/lnInclude
30. }
31. 
32. codeInclude
33. #{
34.    #include "fvCFD.H"
35.    #include <cmath>
36.    #include <iostream>
37. }
38. 
39. }
```